Experimental Validation of Nonsmooth Dynamics Simulations for Robotic Tossing involving Friction and Impacts

Maarten Johannes Jongeneel, Luuk Poort, Nathan van de Wouw, Alessandro Saccon

Experimental Validation of Nonsmooth Dynamics Simulations for Robotic Tossing involving Friction and Impacts

M. J. Jongeneel, L. Poort, N. van de Wouw, and A. Saccon

***Abstract*—In this paper, we evaluate the prediction performance of two nonsmooth rigid-body dynamics simulators on real-world data with spatial impacts in the context of robotic tossing and visual tracking. We perform a parameter identification procedure to find the coefficient of friction and restitution of different objects via a velocity-based and trajectory-based cost function. Our results show that these two identification criteria lead to different parameter values, and these criteria should be chosen in consideration of the application at hand. We compare the simulated predicted rest-pose with measurement data and perform a sensitivity analysis to assess how uncertainty on the identified parameters affects the rest-pose prediction of the object. For the robotic tossing application at hand, we show that the rest-pose prediction is insensitive to the coefficient of restitution, and accurate predictions are obtained via simulations using only ballistic motion and friction.**

***Index Terms*—Motion prediction, Impact, Friction, Contact Modeling, Nonsmooth Mechanics, Parameter Identification.**

Fig. 1: Example of a tossing experiment. In a typical experiment, the box travels about 1.2 meters, impacts, and slides over the conveyor before it comes to rest. A motion capture system records the poses of the box and the conveyor at 360FPS.

I. Introduction

IMPACT-aware robotics is an emerging field of research focusing on the development of robots exploiting physical impacts with objects and environments. In this context, picking and tossing objects using robotic manipulators can be an alternative to traditional pick-and-place solutions. Not only can robotic tossing result in faster object handling, but it also allows robots to place objects outside their kinematic reach [1]. Accurate and fast predictions of the tossing outcome are paramount to proper planning and control of the robot. In this work, we focus on robot tossing in logistics related to the recently funded EU project on Impact-Aware Manipulation[1], see also Figure 1 for an example of a toss. We are interested in the prediction capabilities of rigid-body simulators in the context of visual tracking [2] and robot tossing [3]. Although we focus specifically on logistics, we stress that the problem is general, and our results are relevant for other applications, e.g., robot soccer [4] or robot batting [5].

Multibody dynamics simulators can be used to predict the trajectory of the tossed object. Various simulators exist, and Brogliato et al. [6] discuss that, in general, there are three classes of numerical methods for mechanical systems with contact, friction and impact: *Event-driven methods*, *Penalty methods*, and *Time-stepping methods*.

Event-driven methods numerically integrate the smooth motion in between impacts or contact transitions and re-initialize the state if an impact is detected. These methods require knowledge of the exact impact time, which can be hard to predict. Their main drawback is that they cannot handle an accumulation of impacts, which is typically what occurs in the systems we consider.

Penalty methods, also called pseudo-rigid methods, are based on compliant contact models that allow the interpenetration of objects and apply contact forces proportional to the magnitude of penetration. These methods represent impacts as smooth phenomena, which makes it challenging to combine them with nearly rigid contact and discontinuous friction models [7], [8]. Some examples of commercial software based on penalty methods are MuJoCo [9], Drake [10], and PyBullet [11]. In [12], the authors evaluate the prediction performance of these three simulators on real-world datasets for a cube tossed on a plane and a Cassie robot jumping. In [13], the authors propose a method to learn frictional contact behaviors from data, and they test the proposed method on 3D frictional contact scenarios of cube tosses.

Time-stepping methods rely on rigid-body dynamics and ideal contact/impact laws and use a fixed time-step integration scheme. Time-stepping methods are computationally more efficient than event-driven methods and penalty methods and are well-equipped to deal with the accumulation of impacts. However, rigid-body models are known to have various limitations. Especially in the case of hyperstatic systems, the contact forces cannot be computed uniquely [14], [6]. Nevertheless, these methods generally provide good predictions for long-range motion simulation. This is especially the case if one is not interested in details of local collision (or contact) behavior but rather in the effect of the collision on a global motion [6], which applies to our application of robotic tossing. Furthermore, estimating contact parameters for compliant models (stiffness, damping, etc.) can

The authors are with the Department of Mechanical Engineering, Eindhoven University of Technology (TU/e), The Netherlands {m.j.jongeneel, l.poort, n.v.d.wouw, a.saccon}@tue.nl (*Corresponding author: M. J. Jongeneel*)

[1]This work was partially supported by the Research Project I.AM. through the European Union H2020 program under GA 871899.

be challenging in practice [6]. For these reasons, we choose to use time-stepping methods relying on rigid-body dynamics in our simulation environment. Specifically, we will describe the model in Section II and perform simulations using both a MATLAB implementation and commercial software, for which we use Algoryx Dynamics [15], a European company specialized in multibody dynamics with friction and impacts.

Although multibody dynamics simulators are extensively used in robotic applications, only a few works show an empirical evaluation of the impact prediction performance of these simulators. In [16], [17], [18], [19], [12], the authors show an empirical evaluation of commonly used contact models on 2D datasets. Related to our work, Fazeli et al. [7] propose a parameter identification approach for commonly used rigid-body contact models on an experimental dataset in a two-dimensional setting. More recently, in [16], the authors empirically evaluate their proposed framework by predicting planar dice rolls in a two-dimensional environment. In [20], the authors take a different approach by showing how generic Graph Network Simulators can accurately capture rigid-body dynamics. However, to the best of the author's knowledge, no literature exists describing an empirical evaluation of rigid-body dynamics on 3D data with impacts and friction. Therefore, the main focus of this paper is to empirically evaluate the long-range prediction performance of commonly used rigid-body models on 3D impact data with friction. The main contributions of this work are:

- An identification approach to estimate the contact parameters from 3D impact data with friction;
- Empirical evaluation of the long-range prediction performance of rigid-body simulators based on the identified parameters, using real-world data for box tosses on a conveyor;
- Sensitivity analysis to asses how uncertainty on the identified parameters is reflected in the estimation of the rest-pose ;
- Publicly available source code for parameter identification and evaluation of experimental data[2] and datasets containing hundreds of impact events of multiple box tosses [21].

The remainder of this paper has the following structure. In Section II, we discuss the mathematical preliminaries. Section III describes the experimental setup and data collection procedure. In Section IV, we describe the parameter identification procedure and optimization criteria to find the optimum parameter values of the rigid-body contact models. Section V describes how these identified parameters and their uncertainty are used in simulations for the long-range prediction of box tosses. Section VI describes a sensitivity analysis of the parameters by sweeping each parameter individually and evaluating the rest-pose. The paper finishes with a conclusion in Section VII.

[2]The source code is publicly available through https://gitlab.tue.nl/robotics-lab-public/parameter-identification-and-validation.

II. Mathematical Preliminaries

In this section, we first introduce the notation used in this paper. Then, in Section II-B, we cover the main equations of the nonsmooth dynamical model used. In Section II-C, we cover the Algoryx Dynamics physics simulator used as simulation software.

A. Notation and definitions

In this work, we use right-handed coordinate frames, which are indicated with capital letters (A, B, $\dots$) and further specified by indicating their origin ($\mathbf{o}_A$, $\mathbf{o}_B$, $\dots$) and orthogonal unit vectors ($\boldsymbol{x}_A$, $\boldsymbol{y}_A$, and $\boldsymbol{z}_A$ for frame A, $\boldsymbol{x}_B$, $\boldsymbol{y}_B$, and $\boldsymbol{z}_B$ for frame B, $\dots$). A point is indicated with a bold letter such as $\mathbf{p}$ (or $\mathbf{o}$ when corresponding to a coordinate frame's origin). A coordinate vector with respect to a frame of reference is indicated with a left superscript so that, e.g., ${}^{A}\mathbf{p}$ are the coordinates of $\mathbf{p}$ expressed in A and ${}^{B}\mathbf{p}$ are the coordinates of the same point $\mathbf{p}$ but now expressed in B. We write the *state* of the object as $\mathbf{x} = \{\mathbf{H}, \mathbf{v}\} \in \mathrm{SE}(3) \times \mathbb{R}^6$, where

$$\mathbf{H} = \begin{bmatrix} \mathbf{R} & \mathbf{o} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \in \mathrm{SE}(3) \tag{1}$$

is a transformation matrix with $\mathbf{R} \in \mathrm{SO}(3)$ a rotation matrix and $\mathbf{o} \in \mathbb{R}^3$ the frame origin of a frame fixed to the object. Furthermore, we write

$$\mathbf{v}^{\wedge} = \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{\omega} \end{bmatrix}^{\wedge} := \begin{bmatrix} \boldsymbol{\omega}^{\wedge} & \boldsymbol{v} \\ \mathbf{0}_{1\times 3} & 0 \end{bmatrix} \in \mathfrak{se}(3) \tag{2}$$

as a twist, where we use $\wedge$ (hat) to indicate the classical mapping from $\mathbb{R}^3$ to the corresponding 3×3 skew-symmetric matrix in $\mathfrak{so}(3)$. Similarly, we use $\vee$ (vee) as the inverse mapping [22, Chapter 3.2]. Note that both $\mathbf{H}$ and $\mathbf{v}$ in (1) and (2), respectively, are written as matrices in $\mathbb{R}^{4\times 4}$. For further information regarding the notation, we refer the reader to [23].

B. Nonsmooth dynamics model

As a dynamics model, we use the model derived in [2] and employ the freely available MATLAB implementation developed by the first author [24] as our first simulator. For a full derivation of this model, we refer to [25], but briefly restate some of the main equations here for the sake of readability. We write the equations of motion on the level of momenta, to allow for the inclusion of impulsive forces and discontinuities in the friction force, such that

$$\mathbb{M} d\mathbf{v} + \mathbf{v} \bar{\times}^{*} \mathbb{M} \mathbf{v} dt = \mathbf{f} dt + \mathbf{W}_N d\mathbf{P}_N + \mathbf{W}_T d\mathbf{P}_T, \tag{3}$$

$$\dot{\mathbf{H}} = \mathbf{H} \mathbf{v}^{\wedge}, \tag{4}$$

where we omitted the dependencies on time for the sake of brevity. In (3), $\mathbb{M}$ denotes the inertia tensor, $\bar{\times}^{*}$ the dual-cross product on $\mathbb{R}^6$ as in [23], and $\mathbf{f}(t)$ the generalized wrench containing the generalized forces and torques applied to the center of mass of the body (except for the contact forces), which here involve the gravitational forces. Furthermore, $d\mathbf{P}_N$ and $d\mathbf{P}_T$ are the differential measures of the momenta associated to the contact/impact in normal direction and tangential direction, given as $d\mathbf{P}_N = \boldsymbol{\lambda}_N dt + \boldsymbol{\Lambda}_N d\eta$ and $d\mathbf{P}_T = \boldsymbol{\lambda}_T dt + \boldsymbol{\Lambda}_T d\eta$, respectively, and $\mathbf{W}_N$ and $\mathbf{W}_T$ are the matrices containing the generalized force directions [26]. Note that $\boldsymbol{\lambda}_N dt$ and $\boldsymbol{\lambda}_T dt$ correspond to non-impulsive part of the contact/friction momentum while $\boldsymbol{\Lambda}_N d\eta$ and $\boldsymbol{\Lambda}_T d\eta$ correspond to the impulsive part of the contact/friction momentum.

To understand how the impulsive contact/friction forces and the matrices of generalized force directions are related to the state of the object, we consider Figure 2, where frame C defines the location and orientation of the contact surface and this frame is oriented such that the unit vector $\boldsymbol{z}_C$ defines the normal to the plane. We consider the eight vertices of the box to be the only potential contact points and denote them by $\mathbf{p}_i,\ i \in \{1,\ldots,8\}$. The contact point velocities can now be written as

$$ {}^C\dot{\mathbf{p}}_i = \begin{bmatrix} {}^C\mathbf{R}_B & -{}^C\mathbf{R}_B\,{}^B\mathbf{p}_i^{\wedge} \end{bmatrix} {}^B\mathbf{v}_{M,B}, \tag{5} $$

for $i \in \{1,\ldots,8\}$. In 5, ${}^C\mathbf{R}_B$ is the rotation matrix from frame B to frame C, and ${}^B\mathbf{v}_{M,B}$ is the twist expressing the velocity of frame B with respect to frame M, written in terms of frame B. From 5 we can derive the normal contact point velocity as

$$ \gamma_{N_i} = \begin{bmatrix} {}^C\boldsymbol{z}_C^T\,{}^C\mathbf{R}_B & -{}^C\boldsymbol{z}_C^T\,{}^C\mathbf{R}_B\,{}^B\mathbf{p}_i^{\wedge} \end{bmatrix} {}^B\mathbf{v}_{M,B}, \tag{6} $$

where ${}^C\boldsymbol{z}_C^T = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$ and ${}^B\mathbf{p}_i$ are the vertices of the box expressed in the body-fixed frame B. We will use Newton's impact law, given as

$$ \gamma_{Ni}^{+} = -e_N \gamma_{Ni}^{-}, \tag{7} $$

to relate the pre-impact normal velocity γ_{Ni}^{-} to the post-impact normal velocity γ_{Ni}^{+} through the normal coefficient of restitution e_N [26]. We can rewrite (6) as $\gamma_{Ni} = \mathbf{w}_{Ni}^T\mathbf{v} \in \mathbb{R}$, where $\mathbf{w}_{Ni}^T$ is the row-vector containing the force directions of the contact impulses of $\mathbf{p}_i$, as in (3). We introduce the variable $\xi_{Ni} = \gamma_{Ni}^{+} + e_{Ni}\gamma_{Ni}^{-}$ to formulate an impact law that relates the impulsive force Λ_{Ni} to γ_{Ni}^{+} via a complementarity condition. Using the *proximal point formulation* as in [27], this results in

$$ \Lambda_{Ni} = \operatorname{prox}_{\mathcal{C}_N}\left(\Lambda_{Ni} - r\xi_{Ni}\right)\ \forall i \in \mathcal{I}_c, \tag{8} $$

where $\mathcal{I}_c$ is the set of closed contacts, $\mathcal{C}_N = \mathbb{R}^+$, and $r > 0$.

To model dry friction, we will use a set-valued Coulomb friction model as in [26]. We will define the tangential velocity $\boldsymbol{\gamma}_{Ti}$ as the velocity of $\mathbf{p}_i$ with respect to C in x- and y-direction of the contact surface such that $\boldsymbol{\gamma}_{Ti} = \mathbf{w}_{Ti}^T\mathbf{v} \in \mathbb{R}^2$, where $\mathbf{w}_{Ti}^T \in \mathbb{R}^{2\times n}$ is the matrix containing the force directions of the friction forces acting on $\mathbf{p}_i$.

To account for impulsive friction forces associated with a certain amount of restitution, we write the set-valued force laws for Coulomb friction in terms of momenta. Therefore, similar to [28, Section 5.3.5], we introduce the variable

$$ \boldsymbol{\xi}_{Ti} = \boldsymbol{\gamma}_{Ti}^{+} + e_T\boldsymbol{\gamma}_{Ti}^{-} \in \mathbb{R}^2, \tag{9} $$

where e_T denotes the tangential coefficient of restitution. The tangential coefficient of restitution plays a role in frictional contact impulses and restitution in tangential direction occurs for instance in the motion of the Super Ball [28, Section 5.3.5]. As a result, the set-valued force law for $\boldsymbol{\xi}_{Ti}$ and $\boldsymbol{\Lambda}_{Ti}$ becomes

$$ \boldsymbol{\Lambda}_{Ti} = \operatorname{prox}_{\mathcal{C}_{Ti}}\left(\boldsymbol{\Lambda}_{Ti} - r\boldsymbol{\xi}_{Ti}\right) \tag{10} $$

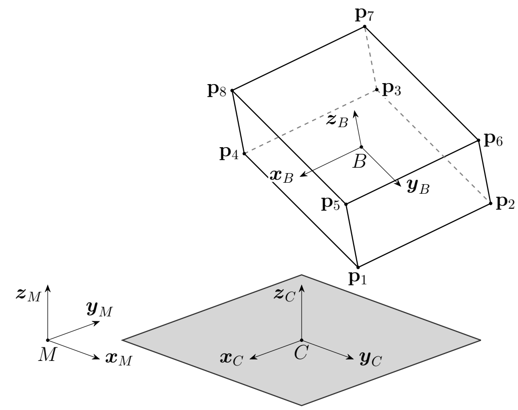


Fig. 2: Definition of frames and points. The Motive frame M, the Conveyor frame C, and the Box frame B are indicated. The contact points of the box are indicated by $\mathbf{p}_i$.

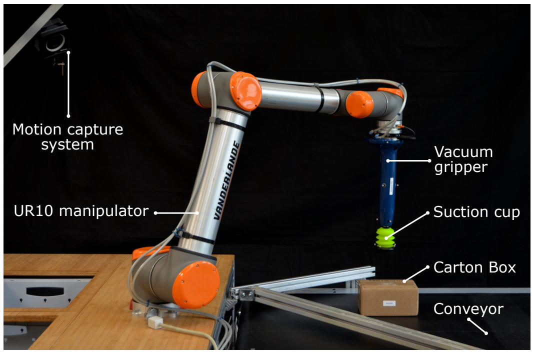
with

$$ \mathcal{C}_{Ti} = \{\boldsymbol{\Lambda}_{Ti} \mid \|\boldsymbol{\Lambda}_{Ti}\| \le \mu\Lambda_{Ni}\} \quad \forall i \in \mathcal{I}_c \tag{11} $$

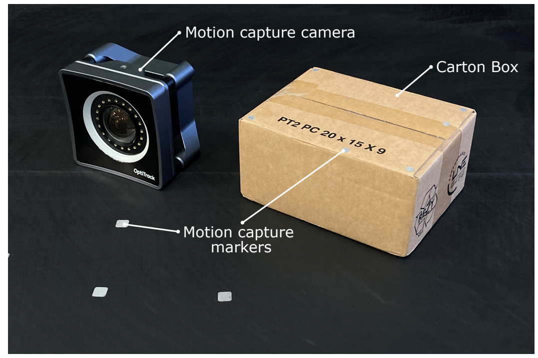
and $r > 0$. Note that the coefficient of tangential restitution e_T, the coefficient of normal restitution e_N, and the coefficient of friction μ appearing in (9), (7), and (11), respectively, depend on both the object and the environment with which it is in contact. These parameters will have to be identified from experimental data, which we will cover in Section IV. In simulations, we set the time step equal to the recording time step of the measurements. Furthermore, we use fixed-point iteration to solve the nonlinear algebraic equations in time-stepping simulation, as in [27]. For further information of modeling rigid-body dynamics with unilateral constraints, the reader is referred to [28], [27], [29], [30], [31], [32].

C. *Algoryx dynamics simulator*

Next to the physics simulator described in the previous section, we will use Algoryx Dynamics [15] as our simulation environment. Algoryx Dynamics is one of the few software packages that are available for simulating nonsmooth mechanics (in contrast to commonly used software packages such as PyBullet, MuJoCo, Drake which treat impacts as smooth phenomena). Although alternatives are possible, such as Siconos [33], we choose to use Algoryx Dynamics as we have a special relation with the developers of Algoryx Dynamics to build a software framework for impact-aware robotics that allows us to integrate our models into their software. In short, Algoryx Dynamics is a software package for modeling and simulating mechanical systems with contact and friction. In our simulations, we use the Iterative Projected Cone Friction model [34, Section 12.15.3.3] where the normal and tangential equations are split. In this model, the normal forces are first solved via a direct solver, and then the normal and tangential equations are solved iteratively. To be able to compare simulations with measurements, we set the simulations time step equal to the recording time step. More details about the time integration methods of Algoryx Dynamics can be found in [35] and further details about the nonsmooth dynamics models of Algoryx Dynamics can be found in [29, Chapter 10].



(a)



(b)

Fig. 3: Experimental Setup. Full setup with the different components indicated (a) and a detailed picture showing, in particular, the suction cup and conveyor markers, the carton box, and the OptiTrack camera (b).

III. Experimental Setup & Data Collection

Figure 3 shows a picture of the experimental setup, which represents an industrial setup used for testing various logistic applications. The setup contains a 6-axis UR10 collaborative robot arm by Universal Robots equipped with an industrial vacuum gripper designed by SmartRobotics[3]. This vacuum gripper has a piGRIP suction cup attached, produced by Piab[4], and can create both an under-pressure and an over-pressure inside the suction cup. The under-pressure allows the robot to pick and hold objects, and the over-pressure speeds up the object release. Furthermore, the setup contains an industrial conveyor belt, and we have conducted experiments with both a moving and stationary conveyor.

Surrounding the setup, an OptiTrack motion capture system (hereafter referred to as mocap) is placed, containing four Prime 17W and two Prime x22 cameras. This system tracks the pose of various objects in the scene at 360 frames per second (fps) using passive reflective markers placed in unique patterns on the objects. Examples of tracked objects are the conveyor belt surface, and multiple boxes. See also Figure 3b. The mocap system assigns a rigid-body frame to each of these objects and expresses the poses of these frames with respect to a reference frame, the origin of the mocap system. Figure 2 shows a schematic where capital letters indicate the frames assigned to the tracked rigid bodies. We refer to the frames as the Conveyor frame (C) positioned at the conveyor belt surface, the Box frame (B) located at the geometric center of the box, and the Mocap frame (M) as the world origin defined by the mocap system.

Figure 4 shows the different boxes used in the experiments and Table I shows their properties, respectively, from left to right. All boxes are uniformly filled with Sculpture Block Foam material (density of 100 kg/m^3), except for Box007, which is uniformly filled with Pinewood (density of 470 kg/m^3). By filling the boxes uniformly, we have a reliable estimate of the boxes' inertia and mass properties and we ensure that the geometric center of the box coincides with the center of mass. The different filling materials allow us to vary in weight, while ensuring all boxes have a mass and dimensions representative for the parcel industry[5]. Note that in Table I, we give the inertia properties with respect to the center of mass, which coincides with the geometric center at which frame B is located. The collected data is made publicly available through the Impact-Aware Robotics Database [36] and all datasets can also be found under the collection in [21].

The experiments involve tosses with a focus on single corner impacts (relevant for parameter identification in Section IV-A) and long-range tosses (relevant for parameter identification and rest-pose prediction in Sections IV-B and V, respectively). The long-range experiments include a phase where the box is released (either by hand or by the robotic arm), a phase with ballistic motion, and a phase where the box impacts the conveyor, before coming to rest after a trajectory of around 1.2 meters. Figure 1 shows an example of a long-range toss where this behavior is also visible. Many long-range tosses have varying initial conditions (release position, orientation, and velocity), which occasionally leads to tumbling of the box upon impact. Generally, tumbling results in unpredictable behavior (think about the toss of a die), which makes predicting the rest-pose of these tosses an unrealistic task. Therefore, experiments where the box tumbles are excluded from the datasets and not considered in this work. Although experiments are executed for multiple boxes, we will only show the results of Box005 and Box006 due to space limitations.

A. Computing body velocities

From our mocap measurements, we obtain the rigid transformation matrices ${}^M\mathbf{H}_B(k)$ and ${}^M\mathbf{H}_C(k)$ in $\mathbb{R}^{4\times4}$ at the discrete-time indices $k \in \{0,\dots,K\}$, where K denotes the total number of discrete time indices of a specific recording. From this data, we compute the velocity using central Euler differencing, which gives, for $k \in \{1,\dots,K-1\}$,

$$ {}^M\dot{\mathbf{o}}_B(k) = \frac{1}{2\Delta\tau}\left({}^M\mathbf{o}_B(k+1) - {}^M\mathbf{o}_B(k-1)\right), \qquad (12) $$

[3]See https://smart-robotics.io/en/

[4]See https://www.piab.com/

[5]This follows from a discussion with Vanderlande (https://www.vanderlande.com/), one of our project partners, providing us with confidential information that cannot be cited here.

(a) (b) (c) (d)

Fig. 4: Four boxes that are considered to be representative of the parcel industry. Box004 (a), Box005 (b), Box006 (c), and Box007 (d). The names correspond to the objects in the Impact-Aware Robotics Database [36] and the box properties are shown in Table I.

TABLE I: Properties of the different boxes of Figure 4. The size is given as {l, w, h} and the inertia is given as $\{I_{xx}, I_{yy}, I_{zz}\}$.

Property	Unit	Box004	Box005	Box006	Box007
Size	mm	{192,85,108}	{230,129,103}	{205,155,100}	{207,158,99}
Mass	kg	0.297	0.566	0.365	1.431
Inertia	kg mm^2	{468,1201,1091}	{1285,2995,3280}	{1040,1590,2020}	{4130,6300,8090}

$$
{}^B\boldsymbol{\omega}^\wedge_{M,B}(k) = \frac{1}{2\Delta\tau}\Bigg(\log\left({}^M\mathbf{R}_B^{-1}(k)\,{}^M\mathbf{R}_B(k+1)\right) - \log\left({}^M\mathbf{R}_B^{-1}(k)\,{}^M\mathbf{R}_B(k-1)\right)\Bigg), \quad (13)
$$

where $\Delta\tau = 1/360$ s is the timestep of the recording. By using

$$
{}^M\boldsymbol{\omega}_{M,B}(k) = {}^M\mathbf{R}_B(k)\,{}^B\boldsymbol{\omega}_{M,B}(k), \quad (14)
$$

and combining (12) and (14), we obtain

$$
{}^{B[M]}\mathbf{v}_{M,B}(k) = \begin{bmatrix} {}^M\dot{\mathbf{o}}_B(k) \\ {}^M\boldsymbol{\omega}_{M,B}(k) \end{bmatrix}, \quad (15)
$$

which is the hybrid body velocity at time k, and represents the twist expressing the velocity of frame B with respect to frame A, written in a frame $B[M]$ whose origin coincides with the origin of frame B and whose orientation coincides with the orientation of frame M. Note that the rotation matrices ${}^M\mathbf{R}_B(k)$ and positions ${}^M\mathbf{o}_B(k)$ follow directly from the transformation matrices ${}^M\mathbf{H}_B(k)$ via (1). In (13), log denotes the matrix logarithm, which can be effectively computed using the inverse of Rodriques formula as in [37, Section 3.2.3.3]. Using the central Euler differencing method to compute the velocities means we effectively use a low-pass filter on our position data, which can be questionable in case of impacts. However, this choice is justified due to our high measurement frequency of 360 frames per second.

IV. Parameter Identification

In this section, we describe the parameter identification procedure. The nonsmooth dynamics model described in Section II-B shows how the coefficient of friction μ, the coefficient of normal restitution e_N and the coefficient of tangential restitution e_T appear in the equations of motion. In previous work (e.g., Fazeli et al. [7]), e_T is not considered, and in our experience [8], e_T has a negligible influence on the prediction accuracy. Therefore, we focus only on identifying μ and e_N and set $e_T = 0$ in all further simulations.

There are different approaches to identifying the coefficient of restitution e_N and the coefficient of friction μ appearing in (7) and (11), respectively. Here, we will consider parameter estimation via a **velocity-based metric** (small time scale) and a **trajectory-based method** (long time scale). Both methods have their pros and cons. The impact law (7) relates the post-impact velocity to the pre-impact velocity via the coefficient of restitution e_N, suggesting that defining a velocity-based metric on a small time window around the impact would make the most sense from an identification criterion point of view. On the other hand, if one is interested in accurately predicting the trajectory or rest-pose of the object (e.g., in our specific case), one might consider defining a trajectory-based metric that minimized the error between the simulated and measured trajectory for a given set of parameters $\{\mu, e_N\}$, where along such a long-range trajectory a multitude of impact/sliding events may take place.

In both cases, we establish a cost function based on the difference between the experimental data and the prediction using a certain parameter set μ, e_N. The parameter values minimizing this cost function therefore result in a prediction that is closest to the experimental data, which we consider ground-truth. Therefore, we define the optimization problem as

$$
(\mu^*, e_N^*) = \arg\min_{\mu, e_N} \frac{1}{M}\sum_{i=1}^{M} L(i; \mu, e_N), \quad (16)
$$

as in [12], subject to $0 \leq \mu$ and $0 \leq e_N \leq 1$, where $L(i; \mu, e_N)$ is the loss function of experiment i depending on the parameter set $\{\mu, e_N\}$, and M is the size of the dataset. Hence, M is the total number of impact events for the velocity-based metric and the total number of trajectories for the trajectory-based metric, which my vary per dataset.

Velocity-based metric: In the velocity-based approach, we consider the pre-impact state of the object as obtained from measurement data, denoted as $\mathbf{x}^- = \{\mathbf{H}^-, \mathbf{v}^-\}$. A trajectory of the box is then simulated with a varying set of contact parameters $\{\mu, e_N\}$ from the initial condition $\mathbf{x}^-$. The loss

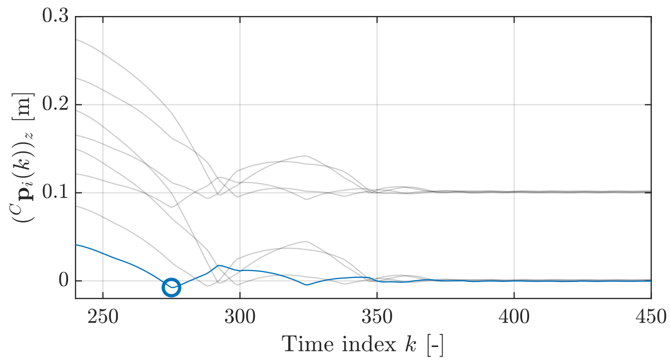


Fig. 5: Normal distance from the eight vertices of a box to the conveyor surface over time for a specific measurement. The blue circle (○) indicates the impact event (at $k = 275$) resulting from the selection scheme of Section IV-A1 and the corresponding trajectory $({}^C\mathbf{p}_1(k))_z$ is highlighted in blue (—). The trajectories of the other vertices $({}^C\mathbf{p}_i(k))_z,\ i \in \{2,\dots,8\})$ are indicted in gray (—).

function computes the error between the post-impact velocity as a result of the simulations (denoted as $\tilde{\mathbf{v}}^+(\mu, e_N)$) with the post-impact velocity as obtained from experimental data (denoted as $\mathbf{v}^+$) for that specific measurement i. Hence, we write the velocity-based loss function as

$$L_{vel}(i; \mu, e_N) = \left\| \mathbf{W}\left(\mathbf{v}^+ - \tilde{\mathbf{v}}^+(e_N, \mu)\right) \right\|_2. \tag{17}$$

To take into account a scaling between the linear and angular velocity components, we use a diagonal weighting matrix $\mathbf{W} = \text{diag}(\begin{bmatrix}1 & 1 & 1 & 0.1 & 0.1 & 0.1\end{bmatrix})$, chosen based on previous experience [8]. The exact definitions of the pre-and post-impact velocity are detailed in Section IV-A2.

Trajectory-based metric: In the trajectory-based approach, we initialize the dynamical model with the state at release $\mathbf{x}(k_{rel})$ and simulate until the box has reached its rest-pose at $\mathbf{x}(k_{rest})$. The moment of release k_{rel} is defined at the moment when the box is released from the robotic arm and is in free flight and the moment of rest k_{rest} is when the relative velocity between the box and the conveyor is zero, see also Section IV-B. Considering (12) - (15), this means that $0 < k_{rel} < k_{rest} < K$. The loss function is based on the approach of [12], where for the varying set of parameter values $\{\mu, e_N\}$, it quantifies the difference between the simulated trajectory $\tilde{\mathbf{H}}(k)$ and the measured trajectory $\mathbf{H}(k)$ at each discrete time index $k_{rel} < k < k_{rest}$. Given the relation between $\mathbf{H}$ and $\mathbf{R}$ and $\mathbf{o}$ in (1), we define the trajectory-based loss function as

$$L_{traj}(i, \mu, e_N) = \frac{1}{N_i}\sum_{k=k_{rel}}^{k_{rest}} \left(\frac{1}{l}\|\mathbf{o}_i(k) - \tilde{\mathbf{o}}_i(k)\|_2 + \|\log\left(\mathbf{R}_i^{-1}(k)\tilde{\mathbf{R}}_i(k)\right)\|_2\right), \tag{18}$$

where $N_i = k_{rest} - k_{rel} + 1$ indicates the total number of discrete time indices considered in experiment i. In (18), the simulated trajectory $\tilde{\mathbf{o}}_i(k), \tilde{\mathbf{R}}_i(k) k \in \{k_{rel}, \dots, k_{rest}\}$, corresponding to measurement i depends on the parameter set $\{\mu, e_N\}$, where the position error is scaled by the length of the box l as in [12].

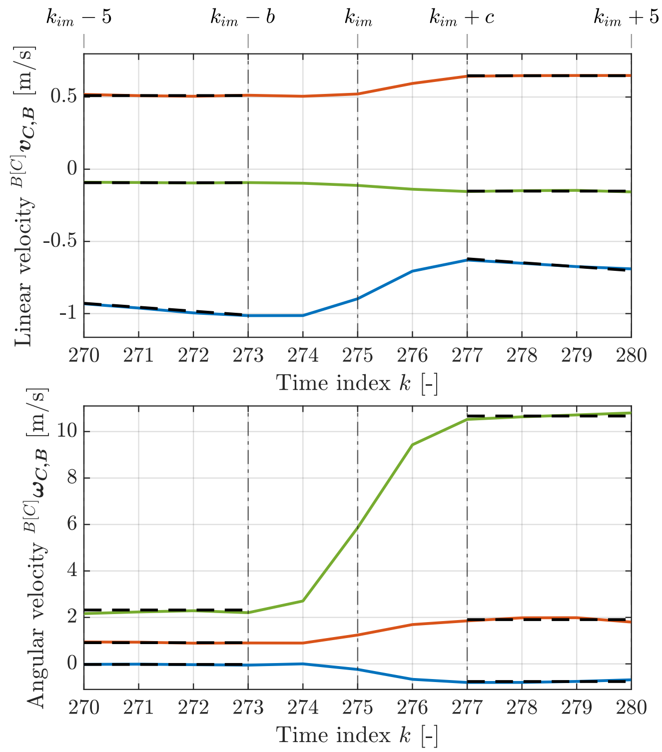


Fig. 6: Linear (top) and angular (bottom) hybrid velocities as result from central Euler differencing of the pose measurements. The x- (—), y- (—), and z- (—) components of the hybrid velocity are indicated by solid lines, while the fitted hybrid velocities are indicated by the black dashed lines (– –). The pre- and post-impact time are indicated by $k_{im} - b$ and $k_{im} + c$, respectively. The data corresponds to the impact event indicated in Figure 5 at the time index $k_{im} = 275$.

In the following subsections, we will further detail the velocity- and trajectory-based parameter identification procedures. For illustration purposes, all data shown in these sections comes from a dataset containing experiments with Box006, see [38]. However, the procedures are performed for all four boxes of Figure 4 and we will present their results by means of error tables.

A. Velocity-based Parameter Identification

The velocity-based approach uses the data about individual impacts to form a cost-function for parameter identification. To this end, experiments are executed where each box is dropped vertically on the conveyor with a single corner impacting the surface. After data is collected, an impact selection procedure selects suitable impact events for parameter identification, from which the pre- and post-impact velocities are computed to subsequently identify the impact parameters.

1) Impact selection procedure: Figure 5 shows the normal distance from the eight corners of a box to the conveyor surface over time for a single. In other words, we have plotted $({}^C\mathbf{p}_i(k))_z$ where $i \in \{1,\dots,8\}$. An impact event induces an inversion of normal velocity, which corresponds to a local minimum in the normal position. As Figure 5 shows, this means that each experiment contains multiple impact events. However, not all impact events are suitable for parameter

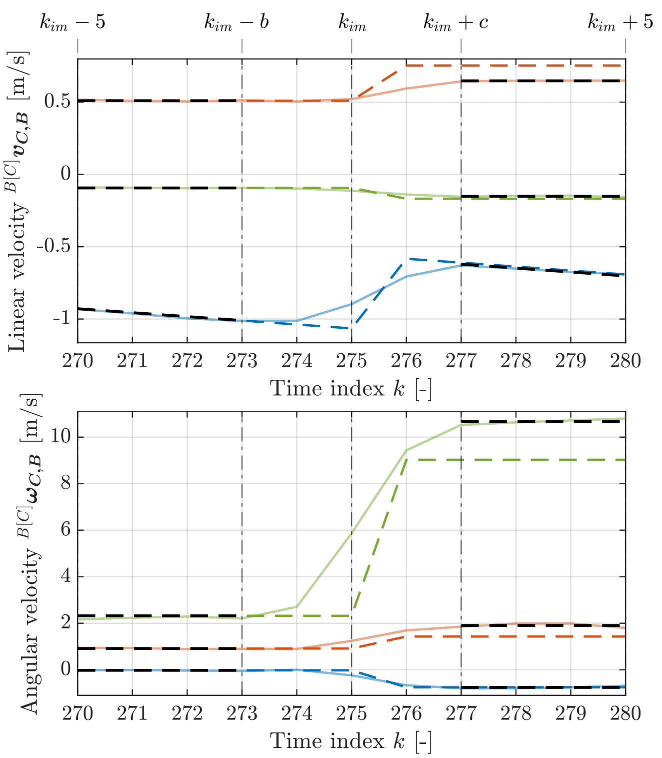


Fig. 7: Linear (top) and angular (bottom) hybrid velocities as result from measurements and simulations with Algoryx Dynamics. We have plotted the x- (—), y- (—), and z- (—) components of the hybrid velocity from measurements and the x- (– –), y- (– –), and z- (– –) components of of the hybrid velocity from simulations with Algoryx Dynamics. The fitted hybrid velocities are indicated by the black dashed lines (– –).

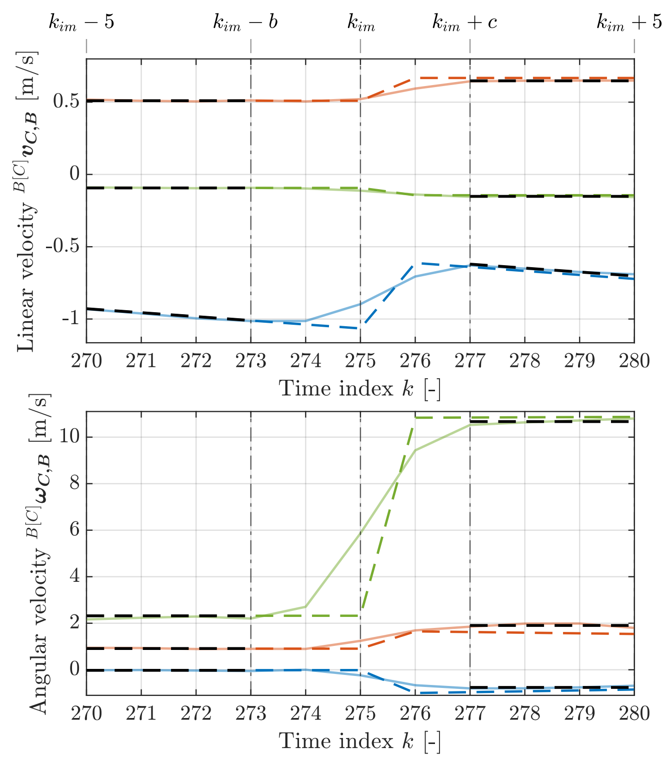


Fig. 8: Linear (top) and angular (bottom) hybrid velocities as result from measurements and simulations with MATLAB. We have plotted the x- (—), y- (—), and z- (—) components of the hybrid velocity from measurements and the x- (– –), y- (– –), and z- (– –) components of of the hybrid velocity from simulations with MATLAB. The fitted hybrid velocities are indicated by the black dashed lines (– –).

identification. More specifically, around each impact event (which has a certain time index k_{im}), we consider a window $\left[k_{im}-5, k_{im}+5\right]$ where the following conditions should hold:

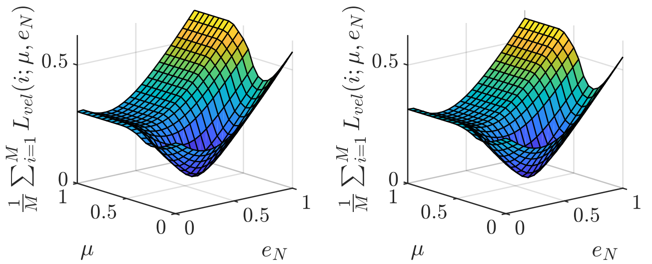
- The impact event cannot be at the beginning or end of the measured data, i.e., $(k_{im}-5) \geq 2$ and $(k_{im}+5) \leq K-1$;
- There can only be one corner of the box making an impact in the time window;
- The normal impact velocity is large enough such that the impact forces dominate other influences such as gyroscopic effects and drag. This minimal impact velocity is set to 0.3 m/s, as this proved to give us the best results.

We subject each impact event to these conditions and remove the ones for which the conditions are not met. For the specific case of Figure 5, this means we find one impact event at $k_{im} = 275$ (indicated by O), corresponding to the first corner of the box.

2) Computing the pre- and post-impact velocity: Figure 6 shows the hybrid body velocities of the box (as described by (15)) for the discrete time window $\left[k_{im}-5, k_{im}+5\right]$ around the impact event of Figure 5. Given the time resolution provided by the mocap system, the impact appears as non instantaneous. To be able to make a fair comparison between the velocity profiles of the measured data and the nonsmooth model (where the impact is instantaneous), we need a clear definition of the pre- and post-impact time index.

Considering Figure 6, we indicate the pre- and post-impact time indices with $k_{im}-b$ and $k_{im}+c$ (see top part of the figure), which in this case correspond to $k_{im}-b = k_{im}-2$ and $k_{im}+c = k_{im}+2$. We identify these time indices by observing that the object is in free flight before $k_{im}-b$ and after $k_{im}+c$. During fee flight, all body velocities are constant, except for the linear velocity in the z-direction (normal to the plane and in the direction of gravity), which is increasing with a constant acceleration, equal to the gravitational acceleration. In the interval $\left[k_{im}-b, k_{im}+c\right]$, we observe a change in velocity as result of the impact, indicating that the impact event takes 4 discrete time steps for the specific impact event shown in Figure 6. However, as each impact event is different, the indices $-b$ and c are carefully selected manually.

With the pre- and post-impact time indices at hand, a straightforward choice would be to take the hybrid body velocities at these time indices as the pre- and post-impact hybrid velocities. However, the body velocities are noisy due to the differentiation of (noisy) position data. Therefore, we use a linear fit over the velocity data in the time windows $[k_{im}-5, k_{im}-b]$ and $[k_{im}+c, k_{im}+5]$ and evaluate these fitted lines at $k_{im}-b$ and $k_{im}+c$ to obtain the hybrid pre- and post-impact velocities, respectively. Note that a linear fit is valid due to the linear influence of the graviational force on the velocity. In the latter, we will refer to the fitted pre- and post-impact hybrid velocities by ${}^{B[C]}\mathbf{v}^{f-}_{C,B}(k_{im}-b)$ and ${}^{B[C]}\mathbf{v}^{f-}_{C,B}(k_{im}+c)$, respectively, with the short hand notation $\mathbf{v}^-$ and $\mathbf{v}^+$.



(a) (b)

Fig. 9: Computed costs for the velocity-based metric as result of simulations with different parameters. Results of Algoryx Dynamics (a) and MATLAB (b) for impact experiments of Box006.

3) Obtaining the velocity-based impact parameters: To compute the optimal impact parameters, we initialize the nonsmooth model of Section II-B and Algoryx dynamics with the fitted hybrid velocity and the pose of the box with respect to the conveyor at $k_{im} - 5$ (beginning of time window). The simulations are then run for the duration of the time-window $[k_{im} - 5,\ k_{im} + 5]$ (11 timesteps) for different parameter values of the parameter set $\{\mu, e_N\}$. More specifically, we vary values for both μ and e_N between 0 and 1 with steps of 0.05. Using smaller steps increases computation time drastically, but does not obtain significantly different results. As a result, we obtain the simulated post-impact velocity at time $k_{im} + c$, denoted as $\tilde{\mathbf{v}}^+$ and compute the loss using (17) for each considered value of μ and e_N. We then sum up all the loss functions for all impact events according to (16). For the specific dataset [38], we have plotted the resulting costs for simulations in Algoryx Dynamics and MATLAB in Figures 9a and 9b, respectively. The costs clearly show an optimum. Furthermore, for $\mu > 0.6$, μ no longer influences the cost. This suggests that for $\mu > 0.6$ the contact is sticking instead of sliding, such that the simulation result is independent of μ. The optimal parameters are selected as those that minimize the cost on the selected grid, which results in $\mu = 0.3$ and $e_N = 0.45$ for MATLAB, and $\mu = 0.25$ and $e_N = 0.4$ for Algoryx Dynamics, for this specific box. In Figures 7 and 8, we show the simulated trajectories for the optimal values using Algoryx Dynamics and MATLAB, respectively, for the same impact event of Figures 5 and 6. We observe that the velocities resulting from simulations closely match the measured velocities, with the results from MATLAB slightly outperforming those of Algoryx Dynamics (considering the y-component of the angular velocity). From Figures 7 and 8, we conclude that, given the optimal contact parameters, the nonsmooth model is able to accurately predict the velocity profiles of a rigid body around the impact time. This can be of particular interest in applications that use a first-step prediction model, such as in visual tracking [2].

B. Trajectory-based Parameter Identification

The trajectory-based parameter identification approach compares simulated and measured trajectories of multiple (different) long-range tosses to find the contact parameters μ and e_N. In a typical experiment, the box travels a total distance of around 1.2 meters, see also Figure 11 for an example

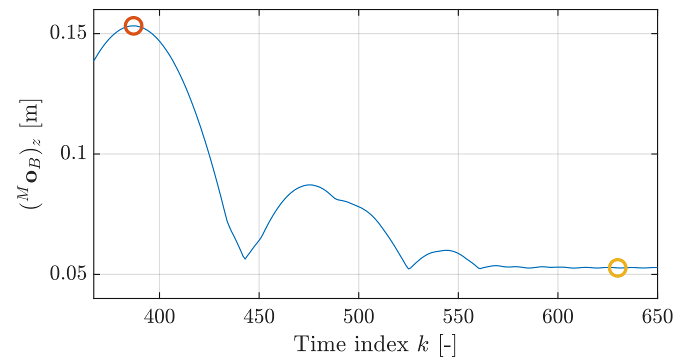


Fig. 10: Indication of the moment of release k_{rel} (○) and the moment of rest k_{rest} (○). The moment of release is the first time index where the box is released and in free flight. The moment of rest is the first time index where the relative velocity between the box and the conveyor is zero. We have plotted $({}^M\mathbf{o}_B)_z$, the z-coordinate of the origin of the box frame B in terms of the inertial frame M (—). Data taken from a dataset with Box006 [39].

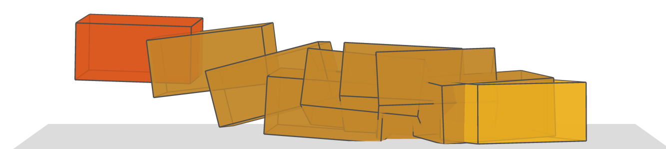
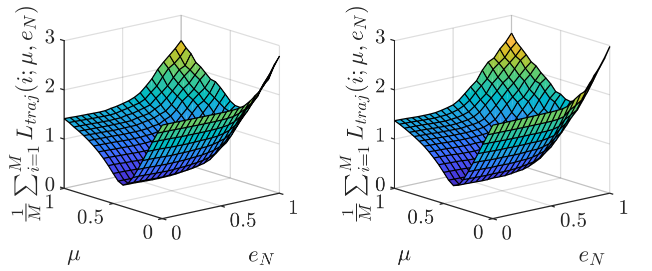


Fig. 11: Measured trajectory of the box of a long-range toss from the moment of release k_{rel} (■) to the moment of rest k_{rest} (■), corresponding to Figure 10. The total travel distance of the box is around 1.2 meters. Data taken from a dataset with Box006 [39].

trajectory of a long range toss with Box006. Figure 1 also shows snapshots of a video of a long range toss. After its release, the box has a ballistic motion, impacts the conveyor, and slides over the conveyor until it comes to rest. Experiments where the box tumbles to another side are excluded from the dataset.

To be able to compare measurements and simulations, we have to define the moment of release and the moment of rest of the box. For illustration purposes, consider Figure 10, where we have plotted $({}^M\mathbf{o}_B)_z$, the z-coordinate of the origin of the box frame B in terms of the inertial frame M, for one specific tossing experiment. In this sequence, we define k_{rel} as the moment of release (indicated by (○) in Figure 10), which is the first time index where the box is in free flight after it is released, and k_{rest} as the moment of rest (indicated by (○) in Figure 10), which is the first time index where the relative velocity between the box and the conveyor is zero. As a result, the total number of discrete time indices $N = k_{rest} - k_{rel} + 1$ is typically different for each experiment. Figure 11 shows the box trajectory of the same experiment in 3D, where it becomes clear that the box slightly bounces upon impact.

For each long-range toss experiment, we initiate a set of simulations with varying parameter values of μ and e_N. We vary these parameters between 0 and 1 with steps of 0.05, as in the velocity-based approach. Furthermore, the measured hybrid velocities ${}^{B[C]}\mathbf{v}_{C,B}(k_{rel})$ and the pose of the box with respect to the conveyor ${}^C\mathbf{H}_B(k_{rel})$ at the time index k_{rel} for that specific experiment are input to the simulations. Each simulation runs from k_{rel} to k_{rest} with a timestep equal to the recording timestep to ensure that measured and simulated trajectories contain the same amount of data points. For each simulated rest-pose (as result of a different value of μ and e_N)



(a) (b)

Fig. 12: Computed costs for the trajectory-based metric as result of simulations with different parameters. Results of Algoryx Dynamics (a) and MATLAB (b) for impact experiments of Box006.

TABLE II: Resulting parameters found for velocity-based and trajectory-based parameter identification

	Velocity-based				Trajectory-based			
	MATLAB		Algoryx		MATLAB		Algoryx	
	μ	e_N	μ	e_N	μ	e_N	μ	e_N
Box004	0.60	0.40	0.65	0.35	0.45	0.05	0.45	0.10
Box005	0.45	0.35	0.40	0.30	0.45	0.10	0.40	0.00
Box006	0.25	0.40	0.25	0.40	0.40	0.25	0.40	0.25
Box007	0.40	0.60	0.35	0.55	0.35	0.45	0.35	0.40

we compute the loss according to (18). We average over all the loss functions over all trajectories according to (16), and we have plotted the resulting costs for simulations in Algoryx Dynamics and MATLAB in Figures 12a and 12b, respectively, for the specific dataset using Box006 [39].

For this specific box, the optimum values are $\mu = 0.4$ and $e_N = 0.25$ for both Algoryx Dynamics and MATLAB, as also shown in Table II. Although not shown here, the costs obtained from simulations with the other boxes have similar shapes. For all boxes, we observe that the cost for Algoryx Dynamics and MATLAB are very similar, both in shape and values. Furthermore, the costs show an apparent insensitivity to e_N for $e_N < 0.6$ It is important to note that, in the tosses we consider, the trajectory of the box is dominated by ballistic motion and sliding, and the box does not tumble. As a result, variations in e_N do not significantly change the trajectory of the simulated toss and therefore do not significantly change the cost. It is likely that the costs will have different shapes if one were to consider tosses where bouncing or tumbling of the box occurs.

V. Rest-pose Prediction Performance

We will now use the identified parameters of Table II to test the model predictions on test datasets of long-range tosses where we consider, as a measure of performance, an error that compares the simulated rest-pose with the measured rest-pose of the box on the conveyor. From a logistics application perspective, this is of particular interest as packages often need to be oriented in a certain way on the conveyor in order to be handled by other systems in the warehouse. In the test datasets, the boxes are tossed manually, which naturally creates variation in the tossing trajectories. For each box of Figure 4, a test dataset is created which can be downloaded via [21]. The results presented in this section are from experiments with a stationary conveyor. Experiments with a moving conveyor are also executed and the datasets can be viewed at [21].

As simulation inputs, we use both the hybrid velocity of the box at k_{rel} (as in Section IV-B) and the optimal parameters μ and e_N as listed in Table II. From these simulations, we obtain the trajectory of the box, of which the rest-pose is of particular interest. As tosses where the box tumbles are excluded from the datasets, the surface normal $\boldsymbol{z}_C$ is aligned with the z-axis of the box frame $\boldsymbol{z}_B$ at all experimental rest-poses. The planar rotation of the box on the surface can therefore be determined quite easily from the box's orientation. We use e_{rot} to indicate the orientation error, which is defined as the relative angle around the normal between the measured and the simulated rest orientation. Because the z-axis of the box and conveyor surface are always aligned, we can extract e_{rot} from the relative orientation by using the 4-quadrant arctangent function obtaining

$$ {}^{B}\mathbf{R}_{\tilde{B}} = ({}^{C}\mathbf{R}_{B}^{-1})\,{}^{C}\mathbf{R}_{\tilde{B}}, \tag{19} $$

$$ {}^{B}\mathbf{R}_{\tilde{B}} = \begin{bmatrix} \cos(e_{rot}) & \sin(e_{rot}) & 0 \\ \sin(e_{rot}) & \cos(e_{rot}) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{20} $$

where ${}^{C}\mathbf{R}_{B}$ indicates the measured rest-orientation and ${}^{C}\mathbf{R}_{\tilde{B}}$ indicates the simulated rest-orientation. Furthermore, we define the position error at the time of rest as

$$ e_{pos} := \left\| \left\{ \left| \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \left({}^{C}\mathbf{o}_{B} - {}^{C}\mathbf{o}_{\tilde{B}}\right) \right| \right\} \right\|_2, \tag{21} $$

which is the norm of the absolute position error in x- and y-direction (in the plane of contact). For all experiments in all datasets related to the boxes of Figure 4, we simulate the different tosses in MATLAB and Algoryx Dynamics and compute the orientation and position errors according to (20) and (21), respectively. Table III shows the average position and orientation error (including the standard deviations) for all experiments with all four boxes, where for each box the test dataset contains fifty tosses. For three randomly selected experiments with Box005 and Box006, Figure 13 shows the rest-pose as result of the velocity-based parameters on Box005 (Figure (a), (b), (c)), results of trajectory-based parameters on Box005 (d),(e),(f), results of velocity-based parameters on Box006 (g),(h),(i), and results of trajectory-based parameters on Box006 (j),(k),(l). The measured rest-pose is shown in gray (■), and the simulation results of Algoryx Dynamics and MATLAB are shown in blue (■) and orange (■), respectively. The simulation results related to the experiments with a moving conveyor (not shown here) show errors in the same order of magnitude as those presented in Table III.

Figure 13 clearly shows how the parameters from the velocity-based cost function of Box006 result in poor predictions, while the parameters obtained from the trajectory-based cost function give significantly better results, both for simulations in MATLAB and Algoryx Dynamics. Interestingly, the errors of the resulting predictions for Box005 are of the same order of magnitude for both sets of parameters, as can also be seen in Table III. In general, the velocity-based cost function may result in parameters that locally around the impact can describe the dynamics quite accurately. This can be concluded

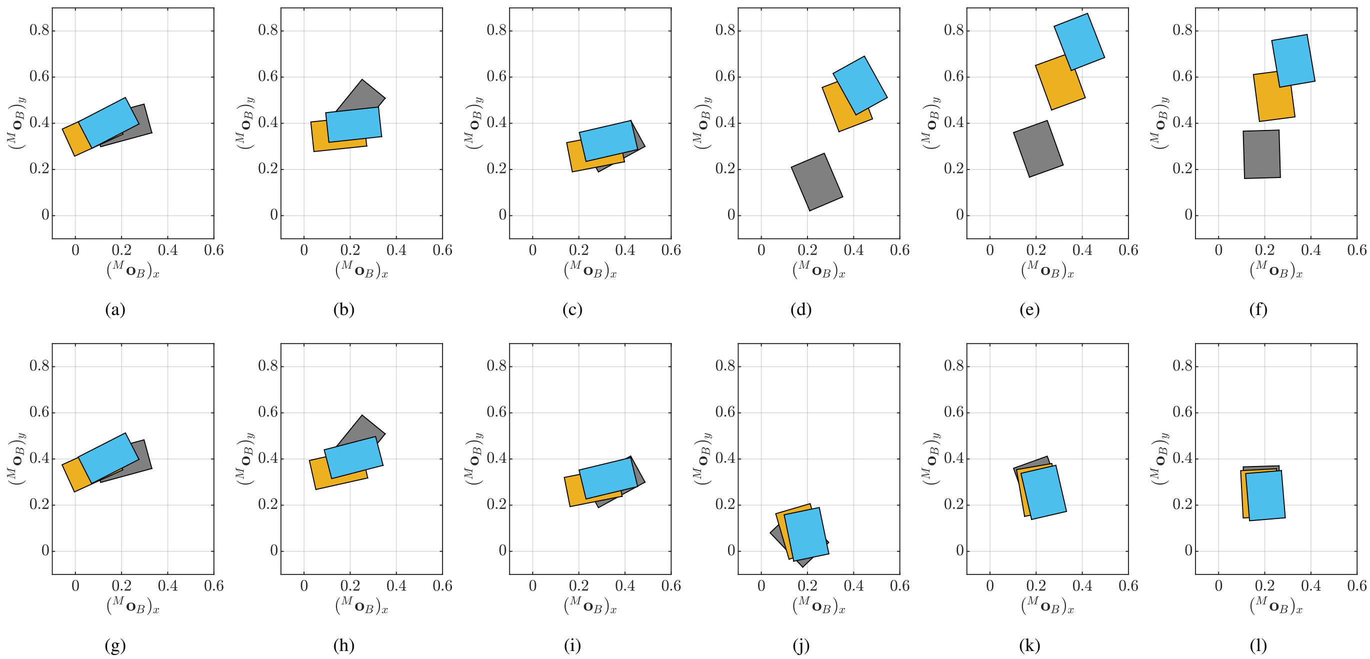


Fig. 13: Resulting rest-poses of three random selected experiments with Box005 and Box006. Results of velocity-based parameters on Box005 (a),(b),(c), results of velocity-based parameters on Box006 (d),(e),(f), results of trajectory-based parameters on Box005 (g),(h),(i), results of trajectory-based parameters on Box006 (j),(k),(l). The measured rest-pose is shown in gray (), and the simulation results of Algoryx Dynamics and MATLAB are shown in blue () and orange (), respectively. Results shown in (j) correspond to the trajectory of Figure 10 and 11.

TABLE III: Errors in the rest-pose prediction from simulations in MATLAB and Algoryx Dynamics, using the parameters for μ and e_N as listed in Table II.

	Box004		Box005		Box006		Box007	
	pos. [cm]	rot. [deg]	pos. [cm]	rot. [deg]	pos. [cm]	rot. [deg]	pos. [cm]	rot. [deg]
vel. based params. MATLAB	19.7 ± 3.5	10.6 ± 8.8	9.9 ± 3.9	12.7 ± 11.3	33.8 ± 8.0	13.2 ± 11.5	6.6 ± 4.3	15.7 ± 17.5
vel. based params. Algoryx	23.4 ± 6.2	11.3 ± 8.3	4.6 ± 2.2	13.6 ± 10.8	49.8 ± 9.2	15.6 ± 14.0	5.0 ± 3.4	16.2 ± 19.6
traj. based params. MATLAB	4.3 ± 2.5	8.2 ± 6.9	10.1 ± 3.7	10.9 ± 8.9	6.7 ± 3.9	16.0 ± 12.3	5.2 ± 3.2	14.1 ± 17.8
traj. based params. Algoryx	6.3 ± 2.9	12.0 ± 8.4	4.6 ± 2.2	10.9 ± 9.3	7.1 ± 3.8	14.5 ± 11.7	4.7 ± 3.2	14.8 ± 18.6

since we observe that the costs of Figures 9a and 9b are close to zero for the optimal parameters. However, if one is interested in the prediction of the rest-pose of the object, the trajectory-based cost function appears to be the better choice. On average, the latter approach results in position errors around five to ten centimeters (which is around 20% to 40% of the maximum box dimension), and orientation errors between ten and sixteen degrees, significantly smaller than the errors obtained using the velocity-based parameters.

VI. Sensitivity Analysis

In this section, we perform a sensitivity analysis in order to understand how uncertainty in the identified parameters reflects on the rest-pose prediction. Therefore, we vary the contact parameters around their optimal values to evaluate the predicted rest-pose sensitivity. Given that the parameters found via the trajectory-based cost function give better long-range prediction accuracy, we will continue using those parameters for the sensitivity analysis. Therefore, in this section we will only consider the parameters listed in the last two columns of Table II.

We perform a sensitivity analysis by changing the value of one of the parameters μ or e_N while keeping the other constant. If we vary a parameter, we consider a range of $[-0.05, 0.05]$ with steps of 0.01 around the optimal parameters listed in Table II. In cases where the optimal value is equal to zero, we consider a range of $[0.00, 0.10]$. Figure 14 shows the results of this sensitivity study for three randomly selected experiments with Box005 and Box006, where the median of the simulation results are shown in full color, and the results of the varied parameters are shown with lower opacity.

Considering these results, we observe that on average, a variation in the coefficient of friction (μ) with ± 0.05 around the mean value, which corresponds to a variation of about $\pm 10\%$, leads to a variation in the rest-pose position of about ± 100mm, which corresponds to a variation of about $\pm 50\%$ of the maximum dimension of the box. At the same time, the variation in the coefficient of friction with ± 0.05 around the mean value leads to a variation in the rest-pose orientation of about ± 3deg. This shows the rest-pose position is rather sensitive the coefficient of friction, while the rest-pose orientation is not. This is also clearly visible in when we consider the corresponding results, shown in the first and third row of Figure 14.

A variation in the coefficient of restitution (e_N) with ± 0.05 around the mean value, corresponding to a variation of about $\pm 10\%$, leads to a variation in the rest-pose position of about ± 4mm, which corresponds to a variation of about $\pm 2\%$ of the

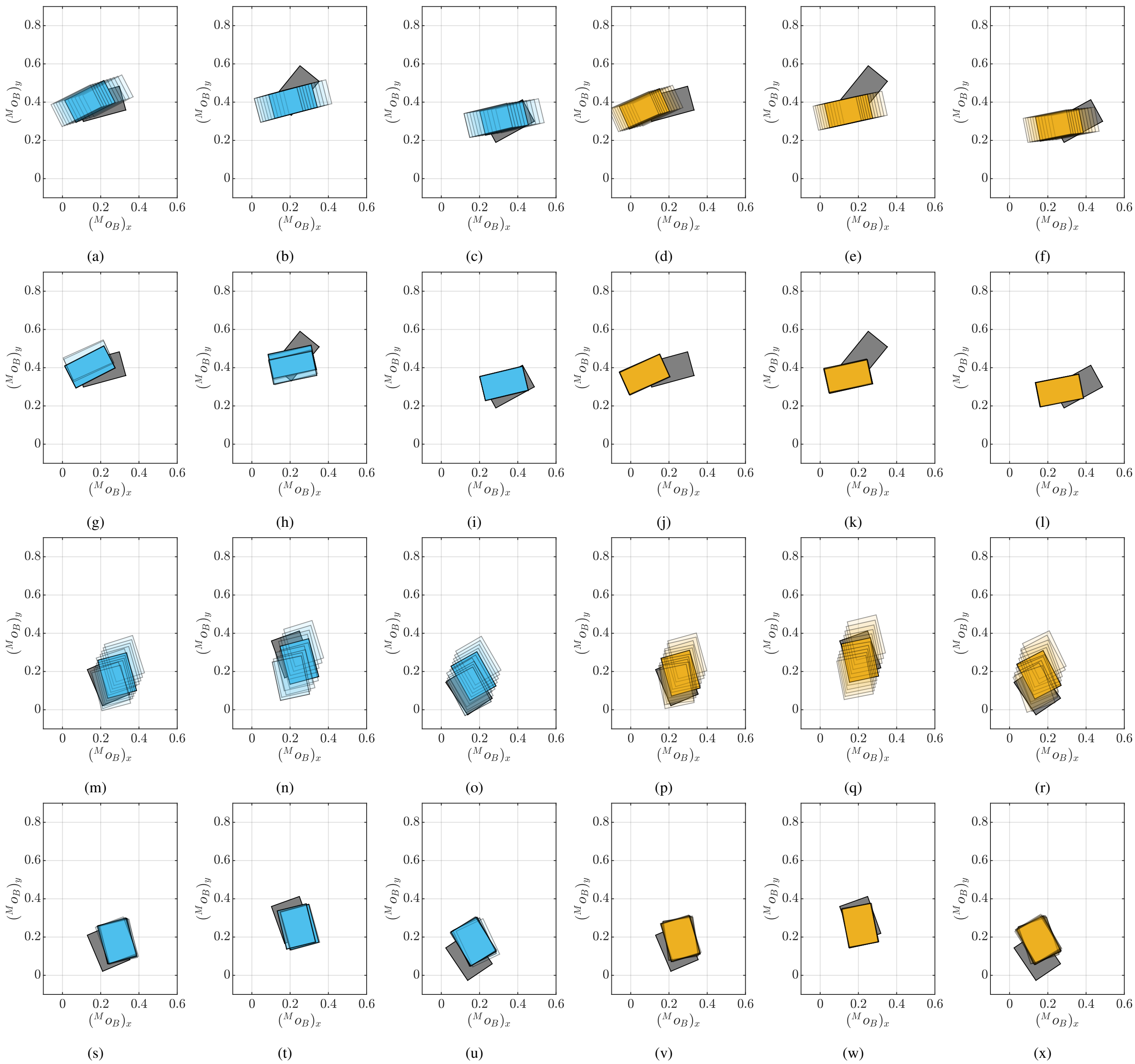


Fig. 14: Resulting rest-poses for varying parameters. First row: result for varying μ for Box005. Second row: results for varying e_N for Box005. Third row: results for varying μ for Box006. Fourth row: results for varying e_N for Box006. The simulation results of Algoryx Dynamics and MATLAB are shown in blue (■) and orange (■), respectively, and the measured rest-pose is in gray (■).

TABLE IV: Errors in the rest-pose prediction from simulations in MATLAB and Algoryx Dynamics. In simulation, we used the trajectory-based parameters of Table II, but setting all values of e_N to zero.

	Box004		Box005		Box006		Box007	
	pos. [cm]	rot. [deg]	pos. [cm]	rot. [deg]	pos. [cm]	rot. [deg]	pos. [cm]	rot. [deg]
traj. based params. MATLAB	4.3 ± 2.5	8.2 ± 7.0	10.1 ± 3.7	10.9 ± 8.5	6.4 ± 3.9	17.6 ± 14.6	4.8 ± 3.1	23.0 ± 19.4
traj. based params. Algoryx	6.2 ± 2.8	10.8 ± 8.6	4.6 ± 2.2	10.9 ± 9.3	7.2 ± 3.7	15.1 ± 11.9	4.5 ± 3.2	21.5 ± 19.2

maximum box dimension. At the same time, the variation in the coefficient of restitution with ± 0.05 around the mean value leads to a variation in the rest-pose orientation of about ± 3deg. This means the rest-pose position and orientation are rather insensitive to a variation in the coefficient of restitution. This is clearly represented by the second and fourth row of Figure 14. These results make sense when considering Figure 12, where both costs (for Algoryx Dynamics and MATLAB) show the insensitivity to a variation in e_N for values of $e_N < 0.6$.

The insensitivity of the rest-pose to a variation in e_N

suggests that if the rest-pose is the main interest, then one can set e_N to zero, simplifying the parameter identification procedure significantly. To test this, we simulate the same set of experiments used in Section V, using the trajectory-based parameters of Table II, but now putting all values of e_N to zero. The resulting errors are shown in Table IV. As expected, the resulting errors are very similar to the ones reported in Table III.

VII. Discussion & Conclusion

This study has empirically evaluated the predictability performance of nonsmooth rigid-body dynamics simulators on experimental test data of box-tosses. We have formulated a parameter identification approach (velocity-based and trajectory-based) to identify the friction and normal-restitution coefficients of different boxes impacting a conveyor surface. These parameters are then used in a predictability performance analysis of rigid-body dynamics simulators, using test data containing 3D impacts with friction. In simulations, we use Algoryx Dynamics and a dedicated dynamics simulator of which the MATLAB implementation is made publicly available.

Considering the parameter identification, we observe that the velocity-based approach leads to a good prediction of the dynamics around the impact event but can lead to poor predictions of long-range tosses. Therefore, we suggest using the velocity-based approach only if one is interested in the local dynamics around the impact event, which can be useful for first-step prediction in applications such as visual tracking. The parameters found via the trajectory-based approach lead to better predictions of the rest-pose, and this approach has our preference, considering the robotic tossing application.

The predictability performance of the simulators is measured in terms of the rest-pose error. These errors show, considering the parameters obtained via the trajectory-based cost function, a position error in the order of 5-10 cm, and a orientation error in the order of 8-16 degrees over a tossing trajectory of about 1.2 meters, with boxes whose maximum dimension is about 20cm. These errors are, considering the application at hand, rather small, and suggest that nonsmooth dynamics models can indeed be used to predict the rest-pose of box-tosses in logistics with sufficient accuracy. The results show that sticking impacts in the experiments lead to larger prediction errors in simulation. Therefore, a more detailed model for the dynamics of sticking impacts is needed to get better simulation predictions.

The sensitivity analysis reveals that the rest-pose prediction is sensitive to the coefficient of friction. On the other hand, the rest-pose prediction is insensitive to e_N as long as its value is below a threshold value (which may vary per box). We show that putting $e_N = 0$ leads to rest-pose prediction errors similar to cases where $e_N \neq 0$. In cases where one is only interested in the rest-pose, as is the case in robotic tossing in logistics, this can simplify the parameter identification procedure. Comparing the prediction performance of nonsmooth dynamics simulators with penalty methods such as PyBullet, MuJoCo, Drake, is left for future work.

Acknowledgment

The authors want to thank Sander Dingemans for his help with the experiments.

References

[1] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser, “TossingBot: Learning to Throw Arbitrary Objects with Residual Physics,” in *Proceedings of Robotics: Science and Systems (RSS)*, 2019.

[2] M. J. Jongeneel, A. Bernardino, N. van de Wouw, and A. Saccon, “Model-Based 6D Visual Object Tracking with Impact Collision Models,” in *IEEE American Control Conference (ACC)*, June 2022, pp. 3850–3856.

[3] M. Lubbers, J. van Voorst, M. J. Jongeneel, and A. Saccon, “Learning Suction Cup Dynamics from Motion Capture: Accurate Prediction of an Object’s Vertical Motion during Release,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2022)*, October 2022.

[4] S. Olufs, F. Adolf, R. Hartanto, and P. Plöger, “Towards Probabilistic Shape Vision in RoboCup: A Practical Approach,” in *RoboCup 2006: Robot Soccer World Cup X*, G. Lakemeyer, E. Sklar, D. G. Sorrenti, and T. Takahashi, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 171–182.

[5] Y.-B. Jia, M. Gardner, and X. Mu, “Batting an in-flight object to the target,” *The International Journal of Robotics Research*, vol. 38, no. 4, pp. 451–485, 2019. [Online]. Available: https://doi.org/10.1177/0278364918817116

[6] B. Brogliato, A. ten Dam, L. Paoli, F. Génot, and M. Abadie, “Numerical simulation of finite dimensional multibody nonsmooth mechanical systems ,” *Applied Mechanics Reviews*, vol. 55, no. 2, pp. 107–150, April 2002.

[7] N. Fazeli, S. Zapolsky, E. Drumwright, and A. Rodriguez, “Fundamental Limitations in Performance and Interpretability of Common Planar Rigid-Body Contact Models,” in *International Symposium of Robotic Research (ISRR)*, Dec. 2017.

[8] L. Poort, “Predictive performance of nonsmooth rigid-body collision models for carton box impacts,” Master’s thesis, Eindhoven University of Technology, Faculty of Mechanical Engineering, Dynamics & Control Section, the Netherlands, 2020, DC 2020.101.

[9] E. Todorov, T. Erez, and Y. Tassa, “Mujoco: A physics engine for model-based control,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.

[10] Russ Tedrake and the Drake Development Team, “Drake: Model-based design and verification for robotics,” 2019. [Online]. Available: https://drake.mit.edu

[11] E. Coumans and Y. Bai, “Pybullet, a python module for physics simulation for games, robotics and machine learning,” 2016–2022. [Online]. Available: http://pybullet.org

[12] B. Acosta, W. Yang, and M. Posa, “Validating Robotics Simulators on Real-World Impacts,” *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6471–6478, 2022.

[13] S. Pfrommer, M. Halm, and M. Posa, “ContactNets: Learning Discontinuous Contact Dynamics with Smooth, Implicit Representations,” in *4th Conference on Robot Learning (CoRL)*, ser. Proceedings of Machine Learning Research, vol. 155, Cambridge, MA, USA, Nov. 2020, pp. 2279–2291.

[14] A. Chatterjee and A. Ruina, “Two Interpretations of Rigidity in Rigid-Body Collisions,” *Journal of Applied Mechanics*, vol. 65, no. 4, pp. 894–900, 12 1998.

[15] Algoryx Simulation AB, “AGX Dynamics,” Umeå, Sweden, 2022, version 2.30.0.0. [Online]. Available: https://www.algoryx.se/agx-dynamics/

[16] N. Fazeli, A. Ajay, and A. Rodriguez, “Long-Horizon Prediction and Uncertainty Propagation with Residual Point Contact Learners,” in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 7898–7904.

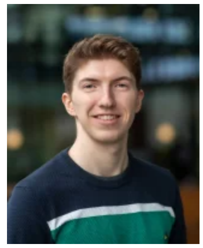
[17] N. Fazeli, E. Donlon, E. Drumwright, and A. Rodriguez, “Empirical evaluation of common contact models for planar impact,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3418–3425.

[18] M. Bauza and A. Rodriguez, “A Probabilistic Data-Driven Model for Planar Pushing,” in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 3008–3015.

[19] R. Kolbert, N. Chavan-Dafle, and A. Rodriguez, “Experimental Validation of Contact Dynamics for In-Hand Manipulation,” in *International Symposium on Experimental Robotics (ISER)*, Tokyo, Japan, 2016.

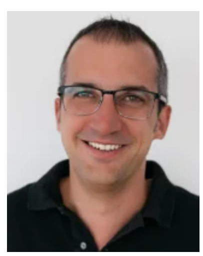
[20] K. R. Allen, T. L. Guevara, Y. Rubanova, K. Stachenfeld, A. Sanchez-Gonzalez, P. Battaglia, and T. Pfaff, "Graph network simulators can learn discontinuous, rigid contact dynamics," in *6th Annual Conference on Robot Learning*, 2022.
[21] M. J. Jongeneel, "Data underlying the publication: Validating rigid-body dynamics simulators on real-world data for robotic tossing applications," *4TU.ResearchData*, Nov 2022, Dataset Collection. [Online]. Available: https://doi.org/10.4121/c.6278310
[22] R. M. Murray, S. S. Sastry, and L. Zexiang, *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Inc., 1994.
[23] S. Traversaro and A. Saccon, *Multibody dynamics notation (version 2)*. Technische Universiteit Eindhoven, Nov. 2019, dC2019.100.
[24] M J Jongeneel, "Box-simulator," Eindhoven, The Netherlands, 2020–2022, version 1.0.0. [Online]. Available: https://github.com/MaartenJongeneel/box-simulator
[25] M. J. Jongeneel, "Model-Based Visual Object Tracking with Collision Models," Master's thesis, Eindhoven University of Technology, Faculty of Mechanical Engineering, Dynamics & Control Section, the Netherlands, 2020, DC 2020.024.
[26] C. Glocker, *Set-Valued Force Laws: Dynamics of Non-Smooth Systems*. Springer-Verlag Berlin Heidelberg, Jan. 2001, vol. 1.
[27] R. Leine and H. Nijmeijer, *Dynamics and bifurcations of non-smooth mechanical systems*, ser. Lecture Notes in Applied and Computational Mechanics. Germany: Springer-Verlag Berlin Heidelberg, 2004, vol. 18.
[28] R. Leine and N. van de Wouw, *Stability and convergence of mechanical systems with unilateral constraints*, ser. Lecture notes in applied and computational mechanics. Germany: Springer-Verlag Berlin Heidelberg, 2008.
[29] C. Lacoursière, "Ghosts and machines : regularized variational methods for interactive simulations of multibodies with dry frictional contacts," Ph.D. dissertation, Umeå University, Computing Science, 2007.
[30] C. Glocker, "Formulation of spatial contact situations in rigid multibody systems," *Computer Methods in Applied Mechanics and Engineering*, vol. 177, pp. 199 – 214, Apr. 1999.
[31] B. Brogliato, *Nonsmooth Mechanics*, 3rd ed. Springer International Publishing Switzerland, 2016.
[32] D. E. Stewart, "Rigid-Body Dynamics with Friction and Impact," *SIAM Rev.*, vol. 42, no. 1, pp. 3–39, Mar. 2000.
[33] V. Acary and F. Pérignon, "Siconos: A Software Platform for Modeling, Simulation, Analysis and Control of Nonsmooth Dynamical Systems," *SNE Simulation News Europe*, vol. 17, no. 3/4, pp. 19–26, Dec. 2007.
[34] A. S. AB, "AGX Dynamics Documentation," 2023. [Online]. Available: https://www.algoryx.se/documentation/complete/agx/tags/latest/doc/UserManual/source/creating_objects.html#iterativeprojectedconefriction
[35] "Time integration method in Algoryx Dynamics," Algoryx Dynamics, 2022, [Accessed 16-Sept-2022]. [Online]. Available: https://www.algoryx.se/documentation/complete/agx/tags/latest/UserManual/source/overall_structure.html#time-integration
[36] A. Saccon and M. J. Jongeneel, "TU/e Impact-Aware Robotics Database," Eindhoven, The Netherlands, 2022, [Accessed 10-Oct-2022]. [Online]. Available: https://impact-aware-robotics-database.tue.nl/
[37] K. M. Lynch and F. C. Park, *Modern Robotics: Mechanics, Planning, and Control*, 1st ed. New York, NY, USA: Cambridge University Press, 2017.
[38] M. J. Jongeneel and S. Dingemans, "Impact Aware Manipulation (I.AM.) archive containing box-drop experiments for Parameter Identification of Box006," *4TU.ResearchData*, Sept. 2022, Dataset. [Online]. Available: https://doi.org/10.4121/21024007.v1
[39] ——, "I.AM. archive containing box-drop experiments for Trajectory Based Parameter Identification of Box006," *4TU.ResearchData*, Oct. 2022, Dataset. [Online]. Available: https://doi.org/10.4121/21387510.v1

Maarten Jongeneel obtained his master's degree in Mechanical Engineering from the Eindhoven University of Technology (TU/e) in 2020. That same year, he started as a doctoral candidate at Eindhoven University of Technology (TU/e) within the European project on Impact-Aware Manipulation (I.AM.), where he is responsible for the modeling and validation of impact models for known robots and objects from motion capture and robot proprioception sensor data. His current research interests are nonsmooth mechanics, impact dynamics, visual object tracking, and multibody dynamics.



Luuk Poort received his M.Sc.-degree (with honors) in Mechanical Engineering at the Eindhoven University of Technology in 2020. He now works as a doctoral candidate on the derivation of modular model reduction techniques and their application to industrial-scale, structural models. His research interests include model reduction, structural dynamics and passivity.

Nathan van de Wouw obtained his M.Sc.-degree (with honours) and Ph.D.-degree in Mechanical Engineering from the Eindhoven University of Technology, the Netherlands, in 1994 and 1999, respectively. He currently holds a full professor position at the Mechanical Engineering Department of the Eindhoven University of Technology, the Netherlands. He has held a (part-time) full professor position the Delft University of Technology, the Netherlands, from 2015-2019. He has also held an adjunct full professor position at the University of Minnesota, U.S.A, from 2014-2021. He is an IEEE Fellow for his contributions to hybrid, data-based and networked control.



Alessandro Saccon is an Assistant Professor in nonlinear control and robotics at the Mechanical Engineering Department, Eindhoven University of Technology (TU/e). His areas of expertise include nonlinear control and estimation, robotics, numerical optimal control and optimization, multi-body mechanics, and geometric mechanics. Alessandro's research interests are focused on modeling, analysis, and control of complex and highly dynamical robotic and mechatronic systems. His current research efforts are directed toward the development and validation of innovative control strategies for robotic systems with multiple intermittent contacts, with application in the field of dynamic robot manipulation and locomotion.