Interval-Arithmetic-Based Trajectory Scaling and Collision Detection for Robots with Uncertain Dynamics

Michael Wagner[1], Stefan B. Liu[1], Andrea Giusti[1,2], and Matthias Althoff[1]
1: Department of Informatics
Technische Universität München, Germany
2: Fraunhofer Italia Research, Italy
Email: [michi.wagner | stefan.liu | andrea.giusti | althoff]@tum.de

Abstract— **We consider two fundamental problems in control of robot manipulators: dynamic scaling of trajectories and collision detection using proprioceptive sensors. While most existing methods approach these problems by assuming accurate knowledge of the robot dynamics, we relax this assumption and account for uncertain model parameters and external disturbances. Our approach is based on the use of a recently proposed interval-arithmetic-based recursive Newton-Euler algorithm. This algorithm enables the efficient numerical computation of over-approximative sets of torques/forces arising from uncertain model parameters. The over-approximative nature of these sets is exploited in this work in order to provide a formally robust trajectory scaling and collision detection strategy. The effectiveness of the proposed approaches has been verified by means of experiments on a 6 degrees-of-freedom robot manipulator with uncertain dynamics.**

I. Introduction

Some of the most relevant techniques currently used in robotics are trajectory scaling and sensorless collision detection. Since most of the available approaches are model-based, their performance relies on accurate models. In this paper, we examine how both techniques can be made robust against inaccurate models, by using a modern recursive algorithm for efficient numerical computation of sets of torques/forces accounting for model uncertainties.

We first review techniques related to the time scaling of trajectories, or *trajectory scaling* in short. In time scaling, a new scaled time is obtained by a function $t' = f(t)$, which ultimately changes the velocity along a given trajectory and thus also the required torques [1]. A requirement in real-world applications is that the exerted torques are always within given constraints. Methods that build upon [1] and merge it with path planning to generate paths that require a minimum amount of time are presented in e.g., [2] and [3]. The authors of [4] present a scheme to dynamically scale a trajectory online with the help of two nested control loops. There, the inner loop ensures that the manipulator follows the trajectory, while the outer loop scales the trajectory so that maximum speed is obtained. An application of trajectory scaling in car manufacturing is proposed in [5], where the authors increase productivity of robotic assembly lines with the help of trajectory scaling.

Next, we review techniques on sensorless collision detection of robot manipulators. This method makes it possible to determine whether a robot collides with an obstacle or itself based on internal sensors only. One of the very first concepts of sensorless collision detection is presented in [6], where the authors treat a collision as a disturbance detected by a disturbance observer; if the disturbance is outside a certain threshold, it can be assumed that a collision has occurred. This disturbance observer assumes the model to be known. An enhancement of this method that relaxes this assumption is presented in [7] by combining sensorless collision detection with parameter estimation: There, the proposed method works in two consecutive phases, where in the first phase the robot identifies the dynamical parameters for a particular trajectory and in the second phase executes that trajectory while being able to detect collision. In [8] the authors first present a concise framework for detecting a collision and then show the various steps needed for a safe reaction movement so that the involved human is not hurt. The authors of [9] propose a projection scheme where they decouple dynamics of the unknown object that the robot handles from the known dynamics of the robot itself. Thus, they can distinguish between a disturbance caused by a collision and one caused by manipulating the unknown object. In [10] the authors present an adaptive scheme to generate dynamic thresholds, so that the collision detection is sensitive to external disturbances and still robust against model uncertainties. The authors of [11] present and experimentally compare several collision detection methods and collision reaction strategies. They also overcome the problem of model uncertainties by combining two different collision detection methods. The resulting detector is then a high-pass filter of the external disturbance. An implementation of a collision detection and reaction scheme on a conventional industrial robot is presented in [12], where the authors build a dynamic collision detection threshold based on the reference trajectory. If the current exceed this threshold, they can distinguish between an intentional contact or an accidental collision, for which they have different reaction strategies. A recent survey about collision detection is presented in [13]. There, the authors compare various collision detection schemes and list their respective advantages and disadvantages. An important remark they make is that “[a] major practical problem is the selection of a threshold on the monitoring signals, so as to avoid false positives and achieve high sensitivity at the same time.” [13, p.2].

Most of the methods mentioned above rely on the assumption that either the model is certain, or that an accurate measurement of the torques or joint accelerations is available. This is a drawback in some practical scenarios, since model parameters, such as the mass associated with the end effector when the robot manipulates various objects, can vary substantially. Furthermore, the required measurements for torque or acceleration are usually noisy or not available at all. We relax this assumption by employing a recent method proposed in [14]: The classical recursive Newton Euler algorithm is enhanced by using interval arithmetic and thus produces over-approximative sets of joint torques/forces that emerge from uncertainties in the dynamical parameters of the model. From now on we refer to this algorithm as the interval-arithmetic-based Newton-Euler algorithm (*IANEA*). A benefit of this technique is that it can compute the torque intervals in a time complexity that is linear in the number of joints and is thus fast enough to be used in real time. We demonstrate how trajectory scaling and collision detection can be solved by using IANEA to handle model uncertainties and input disturbances.

The remainder of this paper is structured as follows: In Sec. II-A the problem of scaling a trajectory under model uncertainties is described in detail and the problem of detecting a collision is explained in Sec. II-B. Subsequently, the proposed methods are described in Sec. III and experimentally verified and discussed in Sec. IV. A conclusion is drawn in Sec. V.

II. Problem Statement

We consider a robot manipulator with N serially connected links, which constitutes an open kinematic chain and is described by the following dynamic model:

$$\begin{aligned}\boldsymbol{\tau} = &\left(\boldsymbol{M}(\boldsymbol{q}) + \boldsymbol{K_r^2 I_m}\right)\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) \\ &+ \boldsymbol{F_v}\dot{\boldsymbol{q}} + \boldsymbol{F_c}\mathrm{sgn}(\dot{\boldsymbol{q}}) + \boldsymbol{d},\end{aligned} \tag{1}$$

where $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}} \in \mathbb{R}^N$ are the joint position, velocity, and acceleration vectors, respectively. $\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{N\times N}$ is the symmetric and positive definite inertia matrix, $\boldsymbol{K_r^2 I_m} \in \mathbb{R}^{N\times N}$ is a diagonal matrix describing the influence of rotor inertias, $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \in \mathbb{R}^N$ returns the vector of Coriolis and centrifugal terms, $\boldsymbol{g}(\boldsymbol{q}) \in \mathbb{R}^N$ is the vector of gravity terms, $\boldsymbol{F_v}\dot{\boldsymbol{q}} \in \mathbb{R}^N$ results in the vector of viscous friction acting on the motors, $\boldsymbol{F_c}\mathrm{sgn}(\dot{\boldsymbol{q}}) \in \mathbb{R}^N$ is the vector of static friction, and $\boldsymbol{d} \in \mathbb{R}^N$ is the vector of external disturbances. Finally, the vector $\boldsymbol{\tau} \in \mathbb{R}^N$ represents the actuation torques/forces.

For brevity we introduce the vector $\boldsymbol{\Delta}$, which contains the dynamical parameters of the model. The dynamical parameters consist, among others, of masses, the coordinates of the center of masses, the barycentric inertia tensors of each link, the viscous and the static frictions acting on the joint motors, the rotor inertias with gear ratios, and the empirically obtained external disturbance torque. These parameters are assumed to be uncertain and bounded.

The trajectory the manipulator has to follow is described by $\boldsymbol{q}_d(t)$, $\dot{\boldsymbol{q}}_d(t)$, and $\ddot{\boldsymbol{q}}_d(t) \in \mathbb{R}^N$, which are the desired joint positions, velocities, and accelerations, respectively. Here, t represents time and is in the interval $t \in [t_0, t_f]$, where t_0 and t_f are the start and end times of the trajectory, respectively.

A. Trajectory Scaling

This subsection provides the problem statement for trajectory scaling. We assume the actuator limits of the robot are known. Therefore, the exerted torque of the i-th motor has to fulfill

$$\tau_i \in [\tau_i^-, \tau_i^+], \tag{2}$$

where τ_i^- and τ_i^+ are the lower and upper torque limits.

We define the duration of the movement as $\Delta t := t_f - t_0$. The framework in [1] describes how a scaled trajectory is obtained by a constant c so that our new duration is $\Delta t_{\mathrm{new}} = \frac{\Delta t}{c}$, and the new trajectory is described by

$$\begin{aligned}\boldsymbol{q}_{\mathrm{new}}(t) &:= \boldsymbol{q}(ct), \\ \dot{\boldsymbol{q}}_{\mathrm{new}}(t) &:= c\dot{\boldsymbol{q}}(ct), \\ \ddot{\boldsymbol{q}}_{\mathrm{new}}(t) &:= c^2\ddot{\boldsymbol{q}}(ct),\end{aligned} \tag{3}$$

where $\boldsymbol{q}_{\mathrm{new}}(t)$, $\dot{\boldsymbol{q}}_{\mathrm{new}}(t)$, and $\ddot{\boldsymbol{q}}_{\mathrm{new}}(t)$ denote the scaled joint position, velocity, and acceleration vectors, respectively. Furthermore, if the torques before scaling can be expressed by

$$\boldsymbol{\tau}(t) = \boldsymbol{M}(\boldsymbol{q}(t))\ddot{\boldsymbol{q}}(t) + \boldsymbol{C}(\boldsymbol{q}(t),\dot{\boldsymbol{q}}(t))\dot{\boldsymbol{q}}(t) + \boldsymbol{g}(\boldsymbol{q}(t)), \tag{4}$$

then the new torques $\boldsymbol{\tau}_{\mathrm{new}}$ are given as

$$\begin{aligned}\boldsymbol{\tau}_{\mathrm{new}}(t) =& c^2\left(\boldsymbol{M}(\boldsymbol{q}(ct))\ddot{\boldsymbol{q}}(ct) + \boldsymbol{C}(\boldsymbol{q}(ct),\dot{\boldsymbol{q}}(ct))\dot{\boldsymbol{q}}(ct)\right) \\ &+ \boldsymbol{g}(\boldsymbol{q}(ct)).\end{aligned} \tag{5}$$

This approach, however, assumes perfect knowledge of the dynamical parameters of the robot. We then face the problem of adapting the trajectory scaling framework to models with uncertain dynamic parameters and ensuring that a resulting trajectory produces only torques that are within the actuator limits for all possible cases within the considered uncertainties.

B. Collision Detection

We next state the problem of sensorless collision detection despite model uncertainties and sensor noise. Sensorless collision detection works by estimating a torque due to external disturbances that arise from collisions. Estimating the torque requires an accurate model and measurements of the trajectory. When this estimated disturbance torque is above a certain threshold, we conclude that a collision has occurred. Finding an appropriate threshold, however, is hard. This threshold has to be as low as possible to detect collisions reliably, yet be high enough as to not falsely recognize torques generated by model uncertainties as collisions.

Now assume that only measurements of joint positions and velocities are available. Furthermore, assume that the dynamical parameters are uncertain. We then face the challenge of finding a function $\boldsymbol{g}$ that generates a torque interval based on measured joint positions, velocities and uncertain dynamical parameters:

$$[\boldsymbol{\tau}] = \boldsymbol{g}\left(\boldsymbol{q}_m, \dot{\boldsymbol{q}}_m, [\boldsymbol{\Delta}]\right). \tag{6}$$

III. Proposed Methods

Both schemes we propose rely on interval arithmetic. It is therefore appropriate to provide some preliminaries. Subsequently, the variables in square brackets denote multidimensional intervals. A multidimensional interval is a set of real numbers defined as $[\boldsymbol{x}] := [\underline{\boldsymbol{x}}, \overline{\boldsymbol{x}}], \underline{\boldsymbol{x}} \in \mathbb{R}^n, \overline{\boldsymbol{x}} \in \mathbb{R}^n, \underline{x}_i \leq \overline{x}_i, \forall i = 1, \dots, n$. The infimum and supremum of $[\boldsymbol{x}]$ are represented by $\underline{\boldsymbol{x}}$ and $\overline{\boldsymbol{x}}$, respectively. Set-based operations are defined as the result of the binary operations $* \in \{+, -, \cdot\}$ of

$$[x] \circledast [y] := \{x * y \,|\, x \in [x], y \in [y]\}. \tag{7}$$

For further details we direct the interested reader to [15].

A. Trajectory Scaling with uncertain dynamics

To obtain a safe scaling factor despite model uncertainties, we propose the idea of first enhancing the standard framework by a more complete model description and then enhancing it to use interval arithmetic.

At first, we extend (5) to account explicitly for motor inertia effects, friction, and disturbance by inserting (3) into (1):

$$\begin{aligned}\boldsymbol{\tau}_{\text{new}}(t) = & c^2 \left((\boldsymbol{M}(\boldsymbol{q}(ct)) + \boldsymbol{K}_{\boldsymbol{r}}^2 \boldsymbol{I}_{\boldsymbol{m}}) \ddot{\boldsymbol{q}}(ct) + \boldsymbol{C}(\boldsymbol{q}(ct), \dot{\boldsymbol{q}}(ct)) \dot{\boldsymbol{q}}(ct) \right) \\ & + c \boldsymbol{F}_{\boldsymbol{v}} \dot{\boldsymbol{q}}(ct) + \boldsymbol{g}(\boldsymbol{q}(ct)) + \boldsymbol{F}_{\boldsymbol{c}} \operatorname{sgn}(\dot{\boldsymbol{q}}(ct)) + \boldsymbol{d}.\end{aligned} \tag{8}$$

Secondly, we adapt (1) to produce torques under model uncertainties. For this we apply interval arithmetic to it which gives us

$$\begin{aligned}[\boldsymbol{\tau}] = & ([\boldsymbol{M}(\boldsymbol{q})] \oplus [\boldsymbol{K}_{\boldsymbol{r}}^2 \boldsymbol{I}_{\boldsymbol{m}}]) \ddot{\boldsymbol{q}} \oplus [\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})] \dot{\boldsymbol{q}} \oplus [\boldsymbol{g}(\boldsymbol{q})] \oplus [\boldsymbol{F}_{\boldsymbol{v}}] \dot{\boldsymbol{q}} \\ & \oplus [\boldsymbol{F}_{\boldsymbol{c}}] \operatorname{sgn}(\dot{\boldsymbol{q}}) \oplus [\boldsymbol{d}].\end{aligned} \tag{9}$$

Considering trajectory scaling as from (4) to (5) applied to (9) results in

$$\begin{aligned}[\boldsymbol{\tau}_{\text{new}}(t)] = & c^2 (([\boldsymbol{M}(\boldsymbol{q}(ct))] \oplus [\boldsymbol{K}_{\boldsymbol{r}}^2 \boldsymbol{I}_{\boldsymbol{m}}]) \ddot{\boldsymbol{q}}(ct) \\ & \oplus [\boldsymbol{C}(\boldsymbol{q}(ct), \dot{\boldsymbol{q}}(ct))] \dot{\boldsymbol{q}}(ct)) \\ & \oplus c [\boldsymbol{F}_{\boldsymbol{v}}] \dot{\boldsymbol{q}}(ct) \\ & \oplus [\boldsymbol{g}(\boldsymbol{q}(ct))] \oplus [\boldsymbol{F}_{\boldsymbol{c}}] \operatorname{sgn}(\dot{\boldsymbol{q}}(ct)) \oplus [\boldsymbol{d}].\end{aligned} \tag{10}$$

For better readability we abbreviate (10) using

$$\begin{aligned}[\boldsymbol{\tau}_\alpha(t)] := & ([\boldsymbol{M}(\boldsymbol{q}(ct))] \oplus [\boldsymbol{K}_{\boldsymbol{r}}^2 \boldsymbol{I}_{\boldsymbol{m}}]) \ddot{\boldsymbol{q}}(ct) \\ & \oplus [\boldsymbol{C}(\boldsymbol{q}(ct), \dot{\boldsymbol{q}}(ct))] \dot{\boldsymbol{q}}(ct) \\ [\boldsymbol{\tau}_\beta(t)] := & [\boldsymbol{F}_{\boldsymbol{v}}] \dot{\boldsymbol{q}}(ct) \\ [\boldsymbol{\tau}_p(t)] := & [\boldsymbol{g}(\boldsymbol{q}(ct))] \oplus [\boldsymbol{F}_{\boldsymbol{c}}] \operatorname{sgn}(\dot{\boldsymbol{q}}(ct)) \oplus [\boldsymbol{d}],\end{aligned} \tag{11}$$

so that

$$[\boldsymbol{\tau}_{\text{new}}(t)] = c^2 [\boldsymbol{\tau}_\alpha(t)] \oplus c [\boldsymbol{\tau}_\beta(t)] \oplus [\boldsymbol{\tau}_p(t)]. \tag{12}$$

Now the goal is to find a value for c so that $\forall t$: $[\tau_{\text{new},i}(t)] \subseteq [\tau_i^-, \tau_i^+], i \in \{1, \cdots, N\}$. Consequently our new torque intervals have to fulfill the following $\forall t$:

$$\forall t, i \in \{1, \cdots, N\} : \underline{\tau}_{\text{new},i}(t) \geq \tau_i^-, \tag{13}$$

$$\forall t, i \in \{1, \cdots, N\} : \overline{\tau}_{\text{new},i}(t) \leq \tau_i^+. \tag{14}$$

Condition (13) leads to

Theorem 1. Given that the following inequalities hold:

$$\begin{aligned}& \frac{-\underline{\tau_\beta}_i(t) + \sqrt{D_i^-(t)}}{2\underline{\tau_\alpha}_i(t)} \leq c_i^-(t) \leq \frac{-\underline{\tau_\beta}_i(t) - \sqrt{D_i^-(t)}}{2\underline{\tau_\alpha}_i(t)}, \\ & \text{if } \underline{\tau_\alpha}_i(t) < 0, \\ & 0 \leq \frac{-\underline{\tau_\beta}_i(t) + \sqrt{D_i^-(t)}}{2\underline{\tau_\alpha}_i(t)} \leq c_i^-(t), \text{ if } \underline{\tau_\alpha}_i(t) > 0,\end{aligned} \tag{15}$$

with $D_i^-(t) := \underline{\tau_\beta}_i(t)^2 - 4\underline{\tau_\alpha}_i(t) \left(\underline{\tau_p}_i(t) - \tau_i^- \right)$ introduced for brevity, (13) and (14) are satisfied. Proof: See Appendix.

The variable $c_i^-(t)$ is then the scaling factor that arises when dealing with joint i and its respective lower actuator limit τ_i^-. Since we want to speed up the trajectory as much as possible, we choose the highest possible value for each c_i^-. The scaling factor $c_i^-(t)$ has to be positive; thus we get

$$c_i^-(t) := \begin{cases} \frac{-\underline{\tau_\beta}_i(t) - \sqrt{D_i^-(t)}}{2\underline{\tau_\alpha}_i(t)}, \text{ if } \underline{\tau_\alpha}_i(t) < 0 \\ \\ \text{NaN}, \text{ if } \underline{\tau_\alpha}_i(t) \geq 0. \end{cases} \tag{16}$$

Here, NaN denotes a value that is not in $\mathbb{R}$ and has the property that

$$\min\{x, \text{NaN}\} = x, \forall x \in \mathbb{R}. \tag{17}$$

Analogously, following condition (14) leads to

$$c_i^+(t) := \begin{cases} \text{NaN}, \text{ if } \overline{\tau_\alpha}_i(t) \leq 0 \\ \\ \frac{-\overline{\tau_\beta}_i(t) + \sqrt{D_i^+(t)}}{2\overline{\tau_\alpha}_i(t)}, \text{ if } \overline{\tau_\alpha}_i(t) > 0, \end{cases} \tag{18}$$

where $D_i^+(t) := \overline{\tau_\beta}_i(t)^2 - 4\overline{\tau_\alpha}_i(t) \left(\overline{\tau_p}_i(t) - \tau_i^+ \right)$. We now seek a constant value for c that is acceptable for all joints at all times. This value is provided by the minimum over all possible valid values. Thus we have

$$c = \min_{i,t} \left\{ c_i^-(t), c_i^+(t) \right\}. \tag{19}$$

To ensure that the result of (19) is not NaN, we first check whether the path is feasible under model uncertainties, i.e., $\underline{\tau_p}_{,i}(t) \geq \tau_i^-, \overline{\tau_p}_{,i}(t) \leq \tau_i^+, \forall t, i \in \{1, \dots, N\}$.

B. Collision Detection

To generate an interval within which the commanded torque must lie, we propose the idea of extending the IANEA from [14] to incorporate an interval for joint accelerations and disturbance and modify it to better handle static friction. When estimating torques based on a trajectory, we face the problem of being unable to reliably measure joint accelerations. We can circumvent this problem by trusting the controller that the difference between the real and desired trajectory will be small, so that the real joint acceleration will be close to the desired acceleration. We therefore treat inaccurate but bounded tracking errors of joint accelerations in the same way as we do for model uncertainties and thus propose a first extension to the IANEA. We assume we

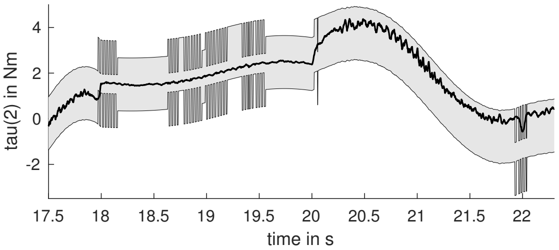

Fig. 1: The chattering effect is visible when velocity is near zero, here shown for axis 2.

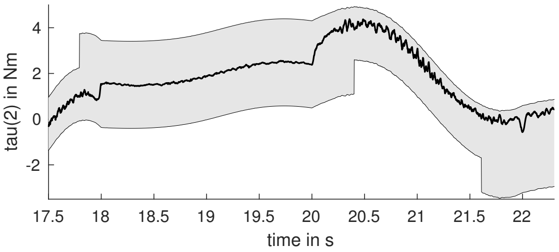

Fig. 2: Chattering effect is removed using the new Coulomb friction model

can estimate the joint acceleration but with disturbance as $\ddot{\boldsymbol{q}}_m = \ddot{\boldsymbol{q}}_d + \boldsymbol{\varepsilon}$, where $\boldsymbol{\varepsilon}$ contains random noise. We cannot know the exact value of $\boldsymbol{\varepsilon}$ but assume that it is bounded, $\boldsymbol{\varepsilon} = [\boldsymbol{\delta}_{\ddot{q}}]$. We then have

$$[\ddot{\boldsymbol{q}}_m(t)] = \ddot{\boldsymbol{q}}_d(t) \oplus [\boldsymbol{\delta}_{\ddot{q}}(t)]. \tag{20}$$

The extension of the IANEA from [14] is straightforward, and every operation that involves $\ddot{\boldsymbol{q}}$ becomes a set-based operation.

A problem we experienced when considering static friction is the chattering in the vicinity of zero velocity. Sensor noise causes the signal to jump frequently between positive and negative values, as shown in Fig. 1. We solve the chattering problem in an over-approximative way by introducing two new interval parameters to the classical Coulomb friction model:

$$[\tau_c] = F_{c,1} \odot sgn(\dot{q} \ominus [F_{c,3}]) \oplus [F_{c,2}], \tag{21}$$

where $F_{c,1}$ is the classical Coulomb friction parameter, $[F_{c,2}]$ is an interval that accounts for friction torque uncertainties, and $[F_{c,3}]$ is an interval that accounts for velocity uncertainty near zero, as illustrated in Fig. 3. For sufficiently large $[F_{c,2}]$, chattering disappears as shown in Fig. 2.

Another problem we experienced, is high-frequency noise in the measured current signal, as generically shown in Fig. 4. This noise is erroneously detected as collisions. To reduce that noise, we low-pass filter the current signal. This, however, introduces the problem that the current is out of phase with respect to the thresholds, as exaggeratedly shown in Fig. 5, and therefore, again, collisions were falsely detected. To compensate the phase shift of the filter, we apply the same filter to $\underline{\boldsymbol{\tau}}$ and $\overline{\boldsymbol{\tau}}$, as illustrated in Fig. 6.

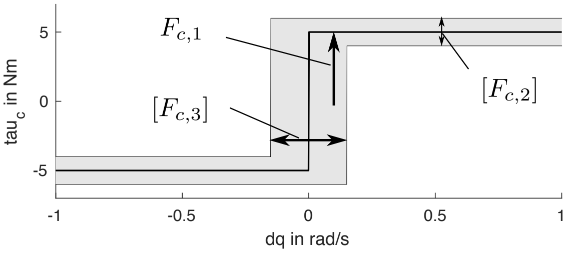

Fig. 3: Coulomb friction model with new interval parameters (gray area)

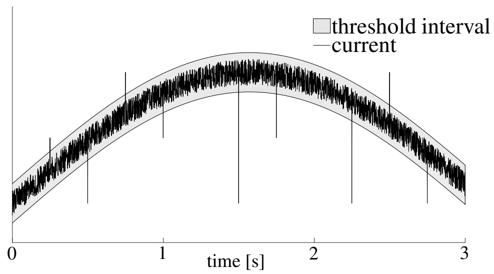

Fig. 4: The measured current has high-frequency noise, that exceeds the thresholds.

With these assumptions in place we can generate a threshold interval online with

$$\begin{aligned} [\boldsymbol{\tau}] =& \text{IANEA}(\boldsymbol{q}_m, \dot{\boldsymbol{q}}_m, [\ddot{\boldsymbol{q}}_m], [\boldsymbol{\Delta}]) \oplus [\boldsymbol{d}] & (22)\\ =& \big([\boldsymbol{M}] \oplus [\boldsymbol{K}_r^2 \boldsymbol{I}_m]\big) \odot [\ddot{\boldsymbol{q}}_m] \oplus [\boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}})] \oplus [\boldsymbol{d}]. & (23) \end{aligned}$$

IV. Evaluation

In this section the proposed methods are experimentally evaluated. First, our method for trajectory scaling is evaluated and subsequently the outcome of the collision detection scheme is examined. The experiments are carried out on a Schunk LWA-4P robot, a light-weight manipulator with 6 revolute joints, using MATLAB R2015b and SIMULINK REAL-TIME running on a Speedgoat Intel® Core™i7-3770K 3.5 GHz. The controller is passivity-based [16, Ch. 7], with a sampling rate of 2 ms. The robot uses electric motors with the property that a commanded current results in a proportional torque. We measure the current; therefore, we work with currents for the remainder of this section.

A. Trajectory Scaling Results

At first, the formula for trajectory scaling under model uncertainties is evaluated. For this experiment the setup is as follows: The currents u_i that can be commanded to joint motors $i \in \{1, \cdots, 6\}$ are restricted to fulfill the conditions $|u_1| \leq 8\,\text{A}$, $|u_2| \leq 8\,\text{A}$, $|u_3| \leq 8\,\text{A}$, $|u_4| \leq 8\,\text{A}$, $|u_5| \leq 2\,\text{A}$, and $|u_6| \leq 2\,\text{A}$. The initial configuration of the manipulator at $t_0 = 0\,\text{s}$ is $\boldsymbol{q}_0 = \begin{bmatrix} \frac{\pi}{4} & -\frac{\pi}{4} & 0 & -\frac{\pi}{4} & \frac{\pi}{2} & -\frac{\pi}{4} \end{bmatrix}^T$rad, and the final configuration at time $t_f = 4\,\text{s}$ is $\boldsymbol{q}_f = \begin{bmatrix} -\frac{\pi}{4} & \frac{\pi}{4} & 0 & \frac{\pi}{4} & -\frac{\pi}{2} & -\frac{3\pi}{4} \end{bmatrix}^T$rad. The trajectory is generated by a 5-th order polynomial. For the unscaled trajectory,

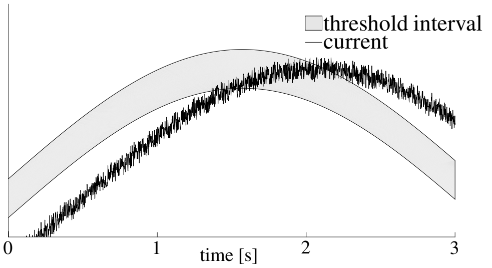

Fig. 5: The filtered current is phase shifted with respect to the thresholds.

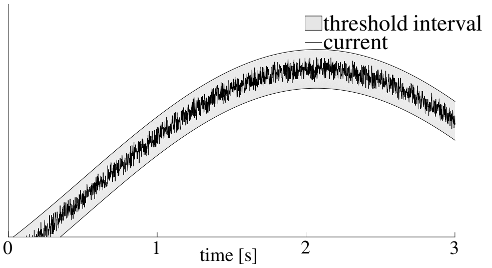

Fig. 6: The measured current and the thresholds are both low pass filtered.

Δt is set to 5 s. The results of the following experiments are shown in Fig. 7. In the first and second rows one can see the measured currents of joint motors 2 and 5, respectively. These two motors are specifically plotted because they are under the most stress when following the trajectories and motor 2 is critical for computing the scaling factor. The third row shows the tracking error when following the trajectory. We define the tracking error $e(t) \in \mathbb{R}$ as

$$e(t) = \|\boldsymbol{q}_m(t) - \boldsymbol{q}_d(t)\| \,. \tag{24}$$

In the first column one can see the results of the unscaled trajectory.

We then scale the trajectory assuming the parameters are certain and that we do not move a payload. For this we use the nominal parameters, which yield a scaling factor of $c = 2.15$ and consequently a Δt_{new} of 2.33 s for our trajectory. The measurements of this experiment are shown in the second column of Fig. 7. As one can see, the current of joint motor 2 saturates between $t = 1\,\text{s}$ and $t = 1.5\,\text{s}$. During this time interval this joint motor can not accurately follow the desired trajectory anymore, and thus the tracking error rises if scaled with the traditional method. This is due to the fact that the nominal model we have is not accurate enough, and as a consequence the traditional method fails.

We let the robot follow the trajectory with the same scaling factor; this time, it manipulates a mass of 1kg with its end effector. As expected, the controller commands more current to the motors to ensure that the robot closely follows the desired trajectory. In the second column, first row, one can see the current commanded to joint motor 2. It is clearly observable that the motor current saturates with the traditional scaling method.

Finally we test our new approach of scaling the trajectory when the robot has an unmodeled payload. We set the lower and upper bounds of the dynamical parameters equal to the nominal parameters, except for the 6-th link. Here we use over-approximative bounds so that its mass can be within an interval of $1.5\,\text{kg}$ to $3.0\,\text{kg}$ and the center of mass can accordingly be shifted between $0\,\text{cm}$ and $4.2\,\text{cm}$ away from its original center of mass. This accounts for a yet unknown mass the robot has to manipulate. We scale the original trajectory using the interval of dynamical parameters and formula (19). This results in a scaling factor of $c = 1.53$ which corresponds to a Δt_{new} of $3.26\,\text{s}$. We then let the robot follow the scaled trajectory while having an additional mass of ca. $1.3\,\text{kg}$ attached to its end effector. This leads to a new effective mass of the end effector of $1.5\,\text{kg} + 1.3\,\text{kg} = 2.8\,\text{kg}$, which is within the bounds of the dynamical parameters. The current commanded to joint motor 2 is plotted in the third column. This time the current is within the limits. Here, it is also observable that with the trajectory generated by the traditional scaling method the currents of both joint motors saturate, which leads to a high tracking error. Compare that to the currents used for the trajectory generated by our new method. Here the currents do not saturate and therefore the tracking error is considerably lower.

B. *Collision Detection Results*

The dynamical parameters are set as follows: The lower and upper bounds are set equal to the nominal model, except for the static friction and the mass parameters of the 6th link. Here we use over-approximative bounds. The interval of static friction ranges from 70% to 130% of the values of the nominal static friction. Furthermore, the values of the inertia tensor and mass range from 100% to 200% of the value of nominal parameters. The center of mass ranges from $-4.2\,\text{cm}$ to $+4.2\,\text{cm}$ around its coordinate center. For the experiments we first let the manipulator follow an arbitrary trajectory without payload and disturb it twice to simulate a collision. The first disturbance is a fast hit at around $t = 2\,\text{s}$ and the second disturbance is a push at around $t = 15\,\text{s}$. The hit and the push are shown in Fig. 8 and Fig. 9, respectively. See Fig. 10 which shows for each axis the commanded current and its respective boundaries. The current stays within the boundaries, except when the robot gets disturbed. Then the current oversteps the thresholds.

Finally we let the robot follow the same trajectory again, this time with a mass attached to its end effector. This leads to an effective mass of the end effector that is within the bounds of the dynamical parameters. We disturb the robot at roughly the same time instances as before. As Fig. 11 shows, the current only oversteps the thresholds at the time instances of the disturbances.

We conclude that a collision has occurred if at least one threshold is violated. Therefore the collision could be detected in both instances.

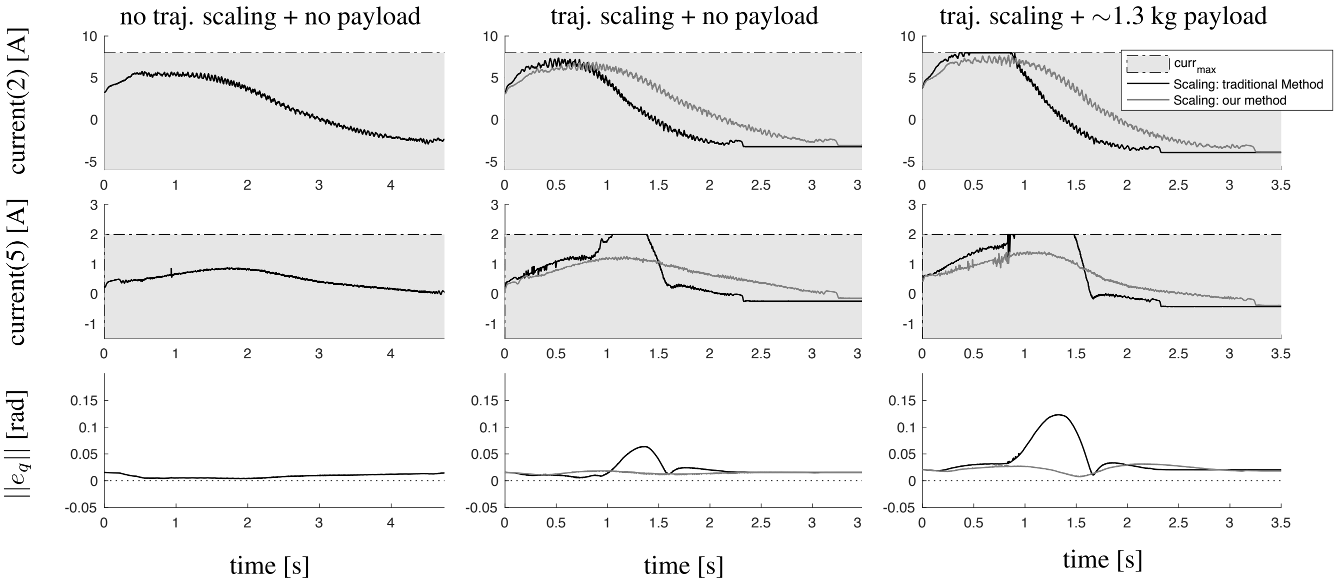

Fig. 7: Results of the trajectory scaling experiments. Saturation of the current is not reached when using our approach, even with payload.

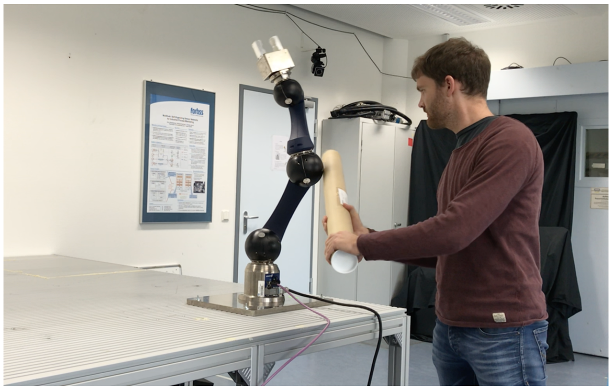

Fig. 8: The robot is hit at joint 3 and 4 (collision 1).

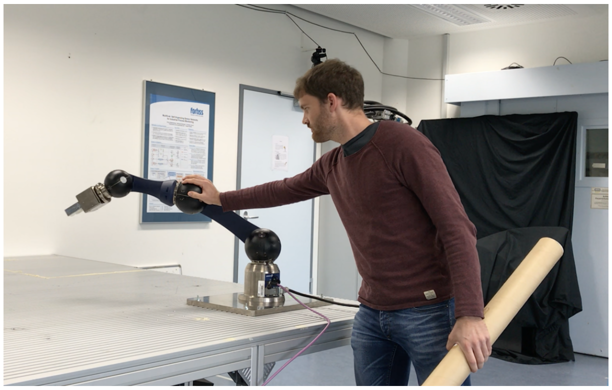

Fig. 9: The robot is pushed at joint 3 and 4 (collision 2).

C. Discussion

The performance of interval-arithmetic-based trajectory scaling and collision detection mainly depends on the amount of uncertainty assumed in the robot model. A large amount of uncertainty in each dynamical parameter decreases the maximum possible scaling factor, as well as the sensitivity of the collision detection in general. Our scheme provides safe scaling despite model uncertainties, so that the current stays within its actuator limits. As demonstrated with the experimental results of trajectory scaling, the tracking error rises when a motor current saturates. Consequently, the robot deviates from the desired path. If the path is relevant, then the consequences can range from loss of performance, to instability in the control of the robot, to collisions. Thus the trajectory must always be scaled so that no current saturates when accurate tracking is required.

The interval-based scheme for sensorless collision detection gives over-approximative bounds for the current which envelopes the measured current. The main benefit of this method is that we have a systematic way of finding thresholds so that no false-positives are detected given the assumed uncertainty bounds.

However, there are as of yet no identification schemes known to the authors that are able to identify uncertainty bounds of each dynamical parameter. One way is to assume that our nominal model is certain (except for friction and payload), such as we have done in this work. We then let the disturbance vector d capture the modeling errors. If the nominal model is accurately estimated using parameter identification schemes, the disturbance level would supposedly stay low. In this work, however, the nominal robot model was found using a CAD model of the robot.

If desired, it is also possible to transform the current signal

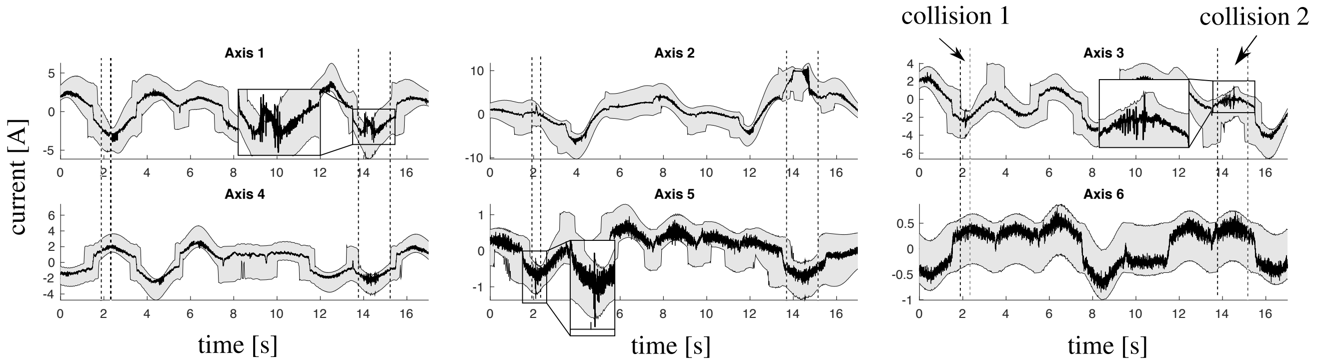

Fig. 10: Interval-arithmetic-based collision detection without payload

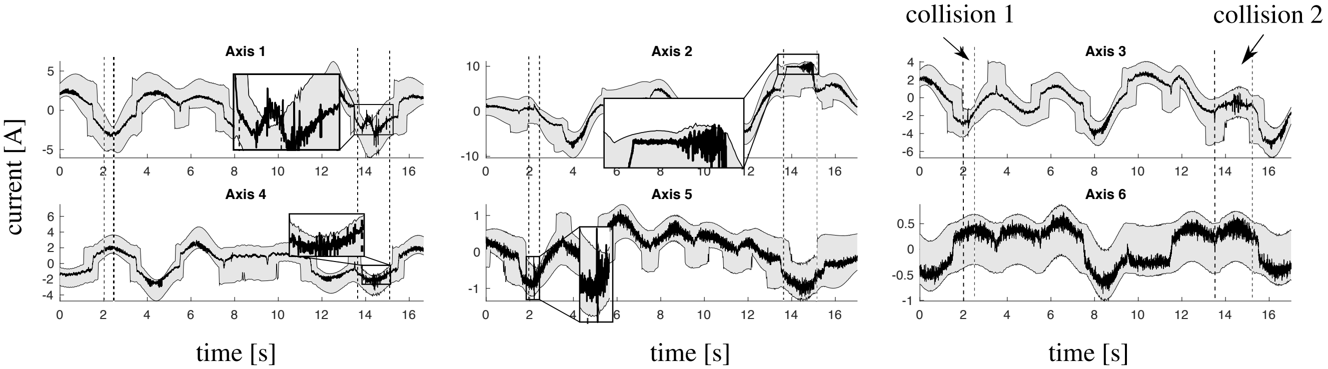

Fig. 11: Interval-arithmetic-based collision detection with payload

into a residual. This residual can be obtained by

$$r = c_m - c_0 \tag{25}$$

$$c_0 = \mathtt{mean}([\mathrm{IANEA}(\boldsymbol{q}_m, \dot{\boldsymbol{q}}_m, [\ddot{\boldsymbol{q}}_m], [\boldsymbol{\Delta}])]), \tag{26}$$

where c_0 is the mean of the interval computed using an uncertain acceleration and the nominal model. The thresholds are transformed accordingly:

$$\theta_{r,up} = \theta_{up} - c_0$$
$$\theta_{r,low} = \theta_{low} - c_0.$$

V. Conclusion

This paper achieves two goals: The first one is to present a method on how to scale a trajectory despite model uncertainties, so that the torques exerted by the joint motors are always within the actuating limits. The experimental results verified the applicability of the proposed approach. This makes it possible to safely speed up a robot movement, even when the dynamical parameters of the robot are not exactly known. For this we simply combined the trajectory scaling framework presented in [1] with the interval-arithmetic-based Newton-Euler algorithm. The second goal is to find a method on how to distinguish a collision from a minor disturbance despite uncertain models. A drawback, however, is that depending on the magnitude of the model uncertainties, the disturbance torque needed to detect a collision can be quite high. In this case the proposed approach is more suitable for detecting self collisions or collisions with the environment, instead.

Appendix

Subsequently we show how (13) leads to (**??**). To prove (**??**) we split (12) into the two equations

$$\underline{\tau}_i = c^2\underline{\tau_\alpha}_{,i} + c\underline{\tau_\beta}_{,i} + \underline{\tau_p}_{,i}, \tag{27}$$

$$\overline{\tau}_i = c^2\overline{\tau_\alpha}_{,i} + c\overline{\tau_\beta}_{,i} + \overline{\tau_p}_{,i}. \tag{28}$$

With (14) we have

$$c^2\underline{\tau_\alpha}_{,i} + c\underline{\tau_\beta}_{,i} + \underline{\tau_p}_{,i} - \tau_i^- \geq 0. \tag{29}$$

To improve readability of the following computations, we define the variables

$$\begin{aligned} x &:= c, \\ A &:= \underline{\tau_\alpha}_{,i}, \\ B &:= \underline{\tau_\beta}_{,i}, \\ E &:= \underline{\tau_p}_{,i} - \tau_i^-, \end{aligned} \tag{30}$$

so that (29) can be written as

$$Ax^2 + Bx + E \geq 0. \tag{31}$$

In the case that $A < 0$ and performing algebraic manipulations, we get

$$\begin{aligned}\left(x+\frac{B}{2A}\right)^2 &\leq \frac{B^2-4AE}{(2A)^2}\\ \Leftrightarrow \left|x+\frac{B}{2A}\right| &\leq \left|\frac{\sqrt{B^2-4AE}}{2A}\right|.\end{aligned}$$

Since $A<0$ it follows that $\frac{+\sqrt{B^2-4AE}}{2A} \leq \frac{-\sqrt{B^2-4AE}}{2A}$, and thus

$$\frac{+\sqrt{B^2-4AE}}{2A} \leq x+\frac{B}{2A} \leq \frac{-\sqrt{B^2-4AE}}{2A}.$$

Solving for x yields

$$\frac{-B+\sqrt{B^2-4AE}}{2A} \leq x \leq \frac{-B-\sqrt{B^2-4AE}}{2A}. \tag{32}$$

In the case that $A>0$, similar algebraic manipulations yield

$$\begin{aligned}\left(x+\frac{B}{2A}\right)^2 &\geq \frac{B^2-4AE}{(2A)^2}\\ \Leftrightarrow \left|x+\frac{B}{2A}\right| &\geq \left|\frac{\sqrt{B^2-4AE}}{2A}\right|.\end{aligned}$$

$$\Leftrightarrow \begin{cases} x+\frac{B}{2A} \leq \frac{-\sqrt{B^2-4AE}}{2A} \leq 0,\\ \text{or}\\ 0 \leq \frac{+\sqrt{B^2-4AE}}{2A} \leq x+\frac{B}{2A}.\end{cases}$$

And therefore

$$\begin{aligned} x &\leq \frac{-B-\sqrt{B^2-4AE}}{2A} \leq 0,\\ &\text{or}\\ 0 &\leq \frac{-B+\sqrt{B^2-4AE}}{2A} \leq x. \end{aligned} \tag{33}$$

With (30), (32), and (33), and the fact that $c>0$ we have

$$0 \leq \frac{-\underline{\tau_\beta}_{,i}+\sqrt{\underline{\tau_\beta}_{,i}^{2}-4\underline{\tau_\alpha}_{,i}(\underline{\tau_p}_{,i}-\tau_i^-)}}{2\underline{\tau_\alpha}_{,i}} \leq c,\ \text{if}\ \underline{\tau_\alpha}_{,i}>0$$

$$\begin{aligned}&\frac{-\underline{\tau_\beta}_{,i}+\sqrt{\underline{\tau_\beta}_{,i}^{2}-4\underline{\tau_\alpha}_{,i}(\underline{\tau_p}_{,i}-\tau_i^-)}}{2\underline{\tau_\alpha}_{,i}} \leq c\\ &\leq \frac{-\underline{\tau_\beta}_{,i}-\sqrt{\underline{\tau_\beta}_{,i}^{2}-4\underline{\tau_\alpha}_{,i}(\underline{\tau_p}_{,i}-\tau_i^-)}}{2\underline{\tau_\alpha}_{,i}},\ \text{if}\ \underline{\tau_\alpha}_{,i}<0\end{aligned} \tag{34}$$

The relationship between (14) and (18) can be followed by analogous reasoning.

ACKNOWLEDGMENT

The research leading to these results has received funding from the European Commission project UnCoVerCPS under grant number 643921.

REFERENCES

[1] J. M. Hollerbach, "Dynamic scaling of manipulator trajectories," *Journal of Dynamic Systems, Measurement, and Control*, vol. 106, no. 1, pp. 102–106, 1984.

[2] B. K. Kim and K. G. Shin, "Minimum-time path planning for robot arms and their dynamics," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-15, no. 2, pp. 213–223, 1985.

[3] G. Sahar and J. M. Hollerbach, "Planning of minimum-time trajectories for robot arms," *The International Journal of Robotics Research*, vol. 5, no. 3, pp. 90–100, 1986.

[4] O. Dahl and L. Nielsen, "Torque-limited path following by online trajectory time scaling," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 5, pp. 554–561, 1990.

[5] D. Meike, M. Pellicciari, G. Berselli, A. Vergnano, and L. Ribickis, "Increasing the energy efficiency of multi-robot production lines in the automotive industry," in *2012 IEEE International Conference on Automation Science and Engineering (CASE)*, 2012.

[6] S. Takakura, T. Murakami, and K. Ohnishi, "An approach to collision detection and recovery motion in industrial robot," in *15th Annual Conference of IEEE Industrial Electronics Society*.

[7] A. D. Luca and R. Mattone, "An adapt-and-detect actuator FDI scheme for robot manipulators," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA 04. 2004*, 2004.

[8] A. Luca, A. Albu-Schaffer, S. Haddadin, and G. Hirzinger, "Collision detection and safe reaction with the DLR-III lightweight manipulator arm," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[9] S.-D. Lee and J.-B. Song, "Collision detection of humanoid robot arm under model uncertainties for handling of unknown object," in *Humanoids*, 2015.

[10] A. Caldas, M. Makarov, M. Grossard, P. Rodriguez-Ayerbe, and D. Dumur, "Adaptive residual filtering for safe human-robot collision detection under modeling uncertainties," *2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pp. 722–727, 2013.

[11] S. Haddadin, A. Albu-Schäffer, A. D. Luca, and G. Hirzinger, "Collision detection and reaction: A contribution to safe physical human-robot interaction," *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3356–3363, 2008.

[12] M. Geravand, F. Flacco, and A. D. Luca, "Human-robot physical interaction and collaboration using an industrial robot with a closed control architecture," in *2013 IEEE International Conference on Robotics and Automation*, May 2013, pp. 4000–4007.

[13] S. Haddadin, A. D. Luca, and A. Albu-Schäffer, "Robot collisions: A survey on detection, isolation, and identification," *IEEE Transactions on Robotics*, vol. PP, no. 99, pp. 1–21, 2017.

[14] A. Giusti and M. Althoff, "Efficient computation of interval-arithmetic-based robust controllers for rigid robots," in *2017 First IEEE International Conference on Robotic Computing (IRC)*, 2017, pp. 129–135.

[15] R. E. Moore, R. B. Kearfott, and M. J. Cloud, *Introduction to Interval Analysis*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2009.

[16] B. Siciliano and O. Khatib, *Springer Handbook of Robotics*, ser. Springer Handbook of Robotics. Springer Berlin Heidelberg, 2008.