An Open-Source ROS-Gazebo Toolbox for Simulating Robots with Compliant Actuators

Riccardo Mengacci [1,*], **Grazia Zambella** [1], **Giorgio Grioli** [2], **Danilo Caporale** [1], **Manuel Catalano** [2] **and Antonio Bicchi** [1,2]

[1] *Research Center "Enrico Piaggio", University of Pisa, Largo Lucio Lazzarino 1, 56126 Pisa, Italy*

[2] *Soft Robotics for Human Cooperation and Rehabilitation, Istituto Italiano di Tecnologia, via Morego, 30, 16163 Genova, Italy*

Correspondence*:
Riccardo Mengacci
riccardo.mengaccigmail.com

ABSTRACT

To enable the design of planning and control strategies in simulated environments before their direct application to the real robot, exploiting the *Sim2Real* practice, powerful and realistic dynamic simulation tools have been proposed, e.g., the ROS-Gazebo framework. However, the majority of such simulators do not account for some of the properties of recently developed advanced systems, e.g., dynamic elastic behaviors shown by all those robots that purposely incorporate compliant elements into their actuators, the so-called Articulated Soft Robots (ASRs). This paper presents an open-source ROS-Gazebo toolbox for simulating ASRs equipped with the aforementioned types of compliant actuators. To achieve this result, the toolbox consists of two ROS-Gazebo modules: a plugin that implements the custom compliant characteristics of a given actuator and simulates the internal motor dynamics, and a ROS manager node used to organize and simplify the overall toolbox usage. The toolbox can implement different compliant joint structures to perform realistic and representative simulations of ASRs, also when they interact with the environment. The simulated ASRs can be also used to retrieve information about the physical behavior of the real system from its simulation, and to develop control policies that can be transferred back to the real world, leveraging the *Sim2Real* practice. To assess the versatility of the proposed plugin, we report simulations of different compliant actuators. Then, to show the reliability of the simulated results, we present experiments executed on two ASRs and compare the performance of the real hardware with the simulations. Finally, to assess the toolbox effectiveness for *Sim2Real* control design, we learn a control policy in simulation, then feed it to the real system in feed-forward comparing the results.

Keywords: Articulated Soft Robots, ROS-Gazebo Simulators, Compliant Actuators, Sim2Real Approach, Digital Twins

1 INTRODUCTION

The applications in which robots have to significantly interact with uncertain environments and to cooperate with the human are growing fast in the last years. One of the major contributions to this growth is given by the recent advances in robotics technologies which see the introduction of structures able to perform such ambitious tasks safely, e.g., by featuring compliance (Albu-Schaffer et al., 2008). There are multiple ways of embedding compliance into the system. One of the possible solutions to this aim is the use of compliant-actuated joints for the robotic system, which form the so-called Articulated Soft Robots (ASRs) (Della Santina et al., 2020). From one side, exploiting the insertion of a constant linear spring into the

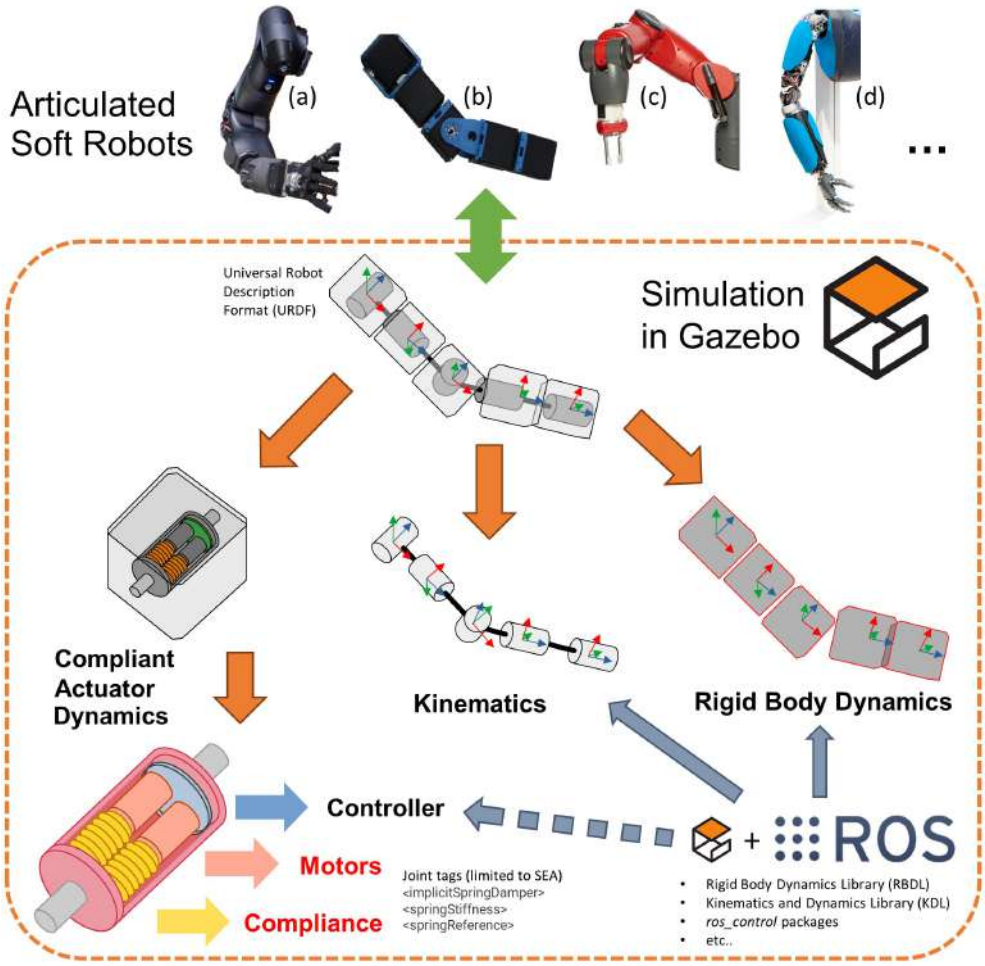

Figure 1. Available solutions for simulating in Gazebo the kinematics and the dynamics of ASRs, such as, (a) WalkMan' arm (Negrello et al., 2015), (b) AlterEgo's arm (Lentini et al., 2019), (c) Baxter's arm (Guizzo and Ackerman, 2012), and (d) Hand-Arm system (Grebenstein et al., 2011). The tools for simulating the compliant-actuated and the motor dynamics are, however, limited to a few simple cases, i.e., the SEA ones. The proposed toolbox allows to reliably simulate also the dynamics of more complex ASRs.

robot's joint, which separates the actuation from the link. This solution is implemented in the Series Elastic Actuators (SEAs) (Pratt and Williamson, 1995). From the other side, more complex and non-linear compliant mechanisms which instead allow modulating the compliance at the joints, namely Variable Stiffness Actuators (VSAs) (Vanderborght et al., 2013), can be realized.

Along with these technological advancements, the availability of effective dynamics simulators facilitated the development of more accurate control and planning strategies of ASRs. This, in turn, accelerated the process of designing and deploying complex compliant-actuated systems into real-world applications. A collection of such simulators is reported by Collins et al. (2021) in a recent survey paper and by Ivaldi et al. (2014). Although many of such simulators are mainly devoted to rigid-body systems, Collins et al. (2021) present also simulators specific for compliant (soft) robots. These simulators, however, do not consider the simplified case of robots equipped with compliant-actuated joints, but rather the case of systems with elastic bodies, i.e., continuum soft robots (Della Santina et al., 2020), which result even more challenging to be modeled and thus require ad-hoc tools. This class goes beyond the purposes of this work and, therefore, will

not be covered. For the rigid-body class, instead, the most popular and widespread open-source dynamics simulators is Gazebo[1], according to Ivaldi et al. (2014), which offers an easy and intuitive connection with the Robotic Operation System[2] (ROS) middleware and the possibility of expanding its functionalities through the use of custom plugins (Cacace et al., 2020). Indeed, leveraging the ROS ecosystem, kinematics and dynamics libraries can be exploited to simulate real systems, as shown in Fig. 1. Despite this, however, the possibility of simulating the complete dynamics of ASRs driven by compliant-actuated joints is not natively included within this powerful tool. In fact, only a few passive characteristics of the joints can be defined, such as the damping, the friction or the stiffness, and the equilibrium position, leveraging on specific URDF tags, while more complex passive mechanisms and possible dynamics of the motor actuation are not considered. The only way to implement such features is to leverage other software, e.g., Matlab/Simulink, or to use custom plugins. To the best of the authors' knowledge, the only example of the latter solution has been presented in (Kameduła et al., 2016), where the authors developed a custom plugin specifically designed for simulating the SEA joints of a centaur-like platform. However, this solution has been only validated through comparative simulations in Matlab and is limited to the SEAs case, thus it neglects most of the other compliant platforms presented in the literature.

Motivated by this and by the need of validating in simulation controllers designed for compliant-actuated VSA structures, e.g., the one proposed in (Zambella et al., 2019), in this paper we present a ROS-Gazebo toolbox that allows simulating ASRs equipped with compliant joints in the class of SEAs and VSAs. The toolbox is composed of two main parts. The first one is used to generate a torque input to the rigid-body dynamics, leveraging the compliance model of the ASRs, and to simulate the dynamics of a direct current (dc) motor, which usually is considered as the actuation source of the compliant joints. The second part, instead, consists of a ROS node used to organize the output of the implementation and to achieve an easy-to-use interface for the overall toolbox. Furthermore, as an additional level of customization, the first part is implemented through a C++ class that can be derived from the users to implement different compliant joint characteristics, according to their platform.

Exploiting this toolbox, ASRs with compliant joints of different types can be accurately modeled and simulated in Gazebo. To this end, we release the implementation of some models relative to the compliant actuators presented by the VIACTORS[3] consortium, which started the investigation of variable stiffness technologies. The proposed toolbox can be easily incorporated in the Universal Robot Description Format (URDF) and, therefore, can serve to speed-up the design of other compliant actuators as well as collect useful and realistic simulation data for the development of planning and control strategies, e.g., for training and learning of specific tasks (Tutsoy et al., 2017). In this regard we show how, with this toolbox, it is possible to rely on simulation data, to design control policies for the real-world systems. As an example of this, we report the results of an Iterative Learning Control (ILC) procedure executed on a simulated platform, and then used to drive the real-world robot, achieving promising performance. In addition to this, simulations on the VIACTORS's compliance models and experimental results on platforms equipped with VSAs are carried out to show the effectiveness of the toolbox and to show how the ASR behaves in simulation. The latter, indeed, we found to be closely comparable to the result obtained from the real applications, even in the case of an interaction scenario.

The paper is organized as follows. The dynamics of ASRs equipped with compliant actuators at the joints is given in Section 2, while the detailed description of the parts which compose the ROS-Gazebo toolbox is

[1] Gazebo: `http://gazebosim.org/`

[2] ROS: `https://www.ros.org/`

[3] VIACTORS: `https://viactors.org/`

reported in Section 3. Simulation and experimental results, which validate and compare the outcome of the toolbox application with results on real platforms, are shown in Section 4. Conclusions are drawn in Section 5.

2 ARTICULATED SOFT ROBOTS DYNAMICS

Leveraging the assumptions stated by Spong (1998), the dynamic model of a multiple degrees of freedom (DOFs) ASR equipped with compliant actuators at the joints can be described by the following contributions (Albu-Schäffer and Bicchi, 2016): the link-side dynamics, the motor-side dynamics and the compliance model that couples these two components.

2.1 Link-side dynamics

The link-side dynamics can be modeled as a rigid-body dynamics supposed driven by a torque source that, in case of ASRs, derives from the elastic mechanism of the compliant actuators. This dynamics is written as

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau_{\mathrm{e},q} + \tau_{\mathrm{ext}} \,, \tag{1}$$

where $q \in \mathbb{R}^n$ is the vector of link positions with its derivatives $\dot{q}, \ddot{q}$, while $M(q), C(q,\dot{q}) \in \mathbb{R}^{n\times n}$ and $G(q) \in \mathbb{R}^n$ are the inertial, Coriolis/centrifugal, and gravitational terms of the system, respectively. The terms $\tau_{\mathrm{e},q} \in \mathbb{R}^n$ are the elastic torque components acting on the link, which depend on the compliance model of the system. Finally, $\tau_{\mathrm{ext}} \in \mathbb{R}^n$ are possible external torques.

2.2 Motor-side dynamics

The motor-side dynamics, instead, depends upon the motors which compose the compliant joints and is computed as

$$B\ddot{\theta} + D\dot{\theta} + \tau_{\mathrm{e},\theta} = \tau_{\mathrm{m}} - \tau_{\mathrm{f}} \,, \tag{2}$$

where $\theta \in \mathbb{R}^m$ is the vector of motor positions with its derivatives $\dot{\theta}, \ddot{\theta}$, and $B, D \in \mathbb{R}^{m\times m}$ are the inertial and damping matrices of the motors, respectively. The term $\tau_{\mathrm{m}} \in \mathbb{R}^m$ is the vector of motor torque inputs, while $\tau_{\mathrm{f}} \in \mathbb{R}^m$ is the torque component due to the presence of friction. $\tau_{\mathrm{e},\theta} \in \mathbb{R}^m$ are the elastic torque components acting on the motors that depend, again, on the compliance model of the system, as detailed in the following. Furthermore, it is worth noting that most of the compliant actuators proposed in the literature are often equipped with an embedded low-level controller that is used to regulate the position of the motors. This basic control, indeed, allows simplifying the use of these devices that can be then regarded as servo-actuators. In this case, the motor torque input is computed e.g., as $\bar{\tau}_{\mathrm{m}}(x,\hat{x})$, where x is the state vector and $\hat{x}$ is the desired state vector used to close the control loop. Therefore, for this case, the motor dynamics can be neglected and the only contribution that drives the link-side dynamics will be the elastic torques.

2.3 Compliance Models

Compliant actuators (Albu-Schaffer et al., 2008) consist of collocated motors θ, typically one or two, connected to the non-collocated link q via an elastic transmission mechanism, as shown in Fig. 2. This implies that the terms $\tau_{\mathrm{e},q}$ and $\tau_{\mathrm{e},\theta}$ in (1) and (2) are such that they couple these two dynamics components to form the complete dynamic model of ASRs.

According to the considered compliant actuator, the characteristics of its elastic mechanism imply different compliance models. As stated in Section 1, among the different types of compliant actuators, the most common are represented by the Series Elastic Actuators (SEAs) and the Variable Stiffness Actuators (VSAs) classes. Furthermore, to the best of the authors' knowledge, almost all the VSAs presented in the literature are always composed of two collocated motors that are the minimum required to regulate either

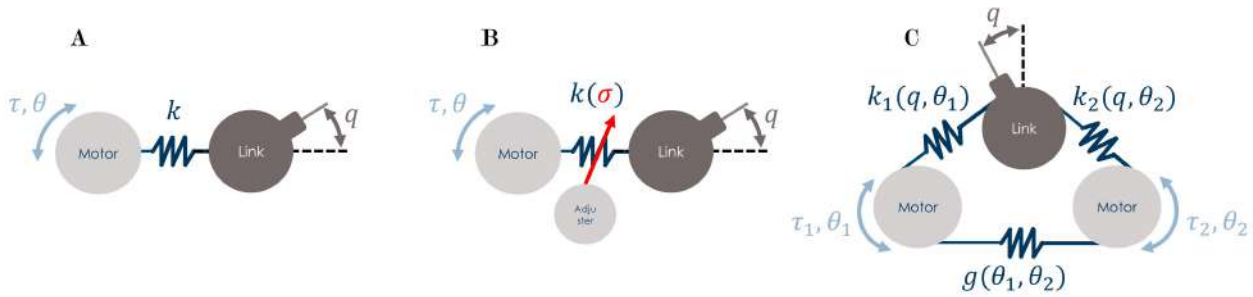

Figure 2. Examples of compliant actuator schemes that can be simulated with the proposed ROS-Gazebo toolbox, (**A**) is a SEA type, while (**B**) and (**C**) are VSAs. The latter two differ for the variable stiffness principle, (**B**) implements a VSA with Adjuster, while (**C**) shows a VSA with Agonistic-Antagonistic (A-A) motor-coupled principle.

the link position and the joint stiffness. Therefore, the case of compliant actuators with more than two collocated motors will not be covered in this work.

The first class consists of an elastic element, i.e., a spring, interposed between the motor and the link, as shown in panel **A** of Fig. 2. In this case, the number of motors of the system is equal to the number of links, i.e., $m = n$, and the spring is characterized by a linear and constant elasticity. The elastic torque is computed as

$$\tau_{\mathrm{e},\theta} = k(q - \theta) = -\tau_{\mathrm{e},q} \,, \tag{3}$$

where the joint stiffness $k \in \mathbb{R}$ is positive and fixed and the difference $\phi = q - \theta$ is named the deflection.

The second class, instead, is realized with more complex and non-linear compliant mechanisms through which the stiffness can be properly modulated. To allow stiffness modulation, the mechanism is equipped with an additional motor, thus the number of motors is twice the number of links, i.e., $m = 2n$. Two main implementation schemes exist in this case: **i**) VSAs with Adjuster (panel **B** in Fig. 2); **ii**) VSA with Agonistic-Antagonistic (A-A) principle (panel **C** in Fig. 2). In the former scheme (**i**), while the first motor is elastically connected to the link, the second one is used to modulate the stiffness directly, typically mechanically, e.g., acting on a pivotal element. The dynamics of the second motor is usually faster than the one of the mechanical system, thus it can be neglected. This results in a kinematic control input for the stiffness variation, often denoted by the symbol σ. Consequently, the elastic torque term is computed as

$$\tau_{\mathrm{e},\theta} = k(\sigma) f(q - \theta) = -\tau_{\mathrm{e},q} \,, \tag{4}$$

where the joint stiffness $k(\sigma) \in \mathbb{R}$ is still positive and can be modulated using the variable σ. Instead, the function $f : \mathbb{R} \to \mathbb{R}$ is typically chosen as a quadratic non-linear function in the deflection ϕ.

Differently, in the latter scheme (**ii**), both motors are used to change the equilibrium position of the link as well as to modulate the joint stiffness, leveraging the agonistic-antagonistic principle. This principle is implemented by a carefully designed non-linear stiffness variation mechanism (Vanderborght et al., 2013). In this case, for each motor i, the elastic torque component is given by

$$\tau_{\mathrm{e},\theta_i} = f_i(q, \theta_i), \quad i = 1, 2 \,, \tag{5}$$

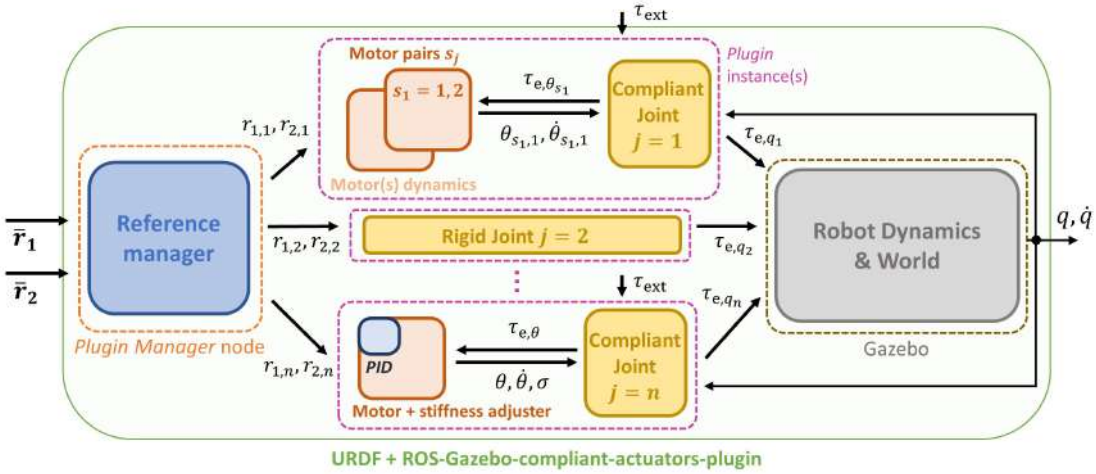

Figure 3. Scheme of the proposed toolbox integrated into the ROS-Gazebo framework, with main state and control variables exchanged between the components of the plugin and the node. Different compliant joints can be simulated as well as rigid ones, with and without motor dynamics and low-level PID control.

where $f_i : \mathbb{R} \to \mathbb{R}$ is a non-linear function of, at least, class $\mathbb{C}^2$. It is worth noting that here the elastic torque seen at the link, differently from the previous cases, is equal to

$$\tau_{e,q} = \tau_{e,\theta_1} + \tau_{e,\theta_2} \ , \tag{6}$$

that is a linear combination of the two elastic torque terms shown in (5). Furthermore, there might be cases for which the two motors have an additional elastic element between them, i.e., there is motor coupling (panel **C** in Fig. 2). In this case, a further term $g(\theta_1, \theta_2)$ must be added (or subtracted) in (5). In addition to this, for the VSA A-A case, leveraging the Equilibrium Point Hypothesis (EPH) state by Feldman (1986), a change of coordinate could be performed to control the two motor positions through two new variables θ_{eq}, θ_{sr}. These variables regulate the equilibrium position of the link and its stiffness preset, respectively, and the motor positions are such that $\theta_1 = \theta_{eq} + \theta_{sr}$ and $\theta_2 = \theta_{eq} - \theta_{sr}$. An example of this control solution is described in Mengacci et al. (2020).

3 ROS-GAZEBO TOOLBOX DESIGN

To simulate the complete ASR dynamics presented in the previous section, obtained by combining (1) and (2), we exploit the ROS middleware with the Gazebo simulation software. More specifically, we leverage the fact that equation (1) is commonly evaluated inside the Gazebo simulator with the support of open-source physic engines such as the Open Dynamic Engine[4] (ODE), starting from the robot's model (i.e., a list of joints and links, with kinematic and dynamic parameters) described in the Universal Robotic Description Format (URDF) file. Regarding the elastic torque components $\tau_{e,q}$ and $\tau_{e,\theta}$, the motor dynamics (2), and the low-level controller, we developed a ROS-Gazebo toolbox which consists of two main parts: i) one C++ plugin class in which the compliance model, as well as the motor dynamics and the controller, are implemented, and ii) a ROS node used to organize the input and output variables of the toolbox. In the following sections, we describe these parts in detail. The overall structure of the proposed toolbox is depicted in Fig. 3.

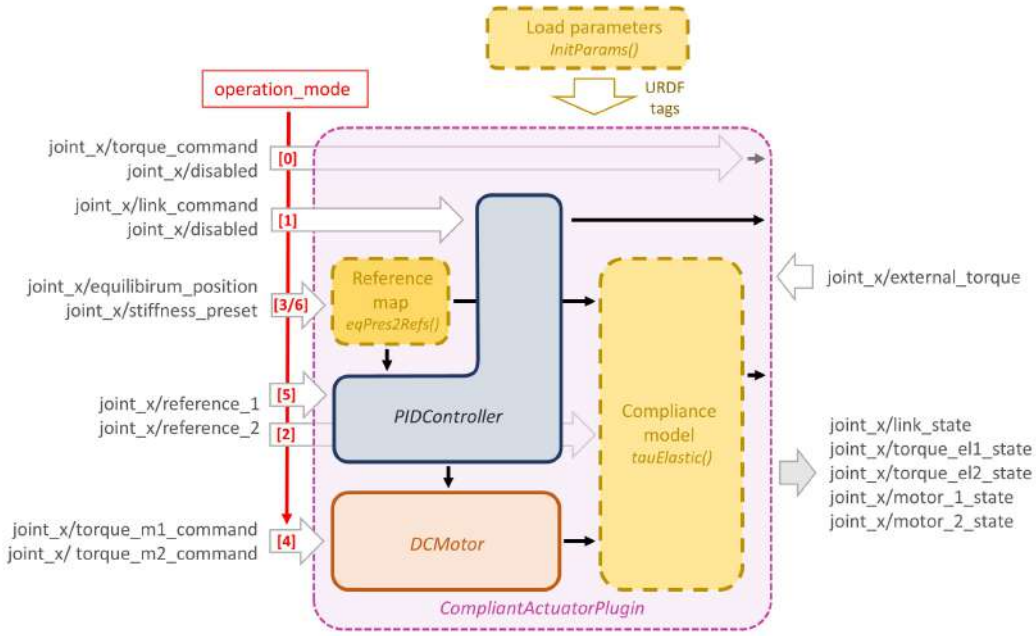

Figure 4. The detailed structure of the *CompliantActuatorPlugin*, with its components and the topic names with which it interacts. The dashed yellow boxes reported in this figure are *virtual* functions (see Section 3.1.3) of the main C++ class that can be customized and should be implemented by the user, while the other blocks are used to simulate the motor dynamics and a simple PID control loop.

3.1 Compliant Actuator Plugin

The purpose of this plugin, implemented as a C++ class, is to define how the elastic torque of the compliant actuator, i.e., $\tau_{e,q}$, is computed and to assign it to the joint torque of the Gazebo model. Furthermore, this plugin implements the dynamics of a direct current (dc) motor and a simple PID controller. The structure of this plugin, namely *CompliantActuatorPlugin*, is shown in Fig. 4 and is composed of different blocks: **a**) the compliance model, in which the elastic functions and their parameters (i.e., $f, f_i, k, k(\sigma)$ in (3), (5) and (4)) are defined; **b**) the *DCMotor* block, that implements the motor dynamics and from which the motor states, e.g., θ_1, θ_2, are retrieved; **c**) the *PIDController* block, where the control law $\bar{\tau}_{\mathrm{m}}(x, \hat{x})$ is implemented, and **d**) the reference map used to perform the change of variables from motor positions to equilibrium position and stiffness preset, i.e., $\theta_{\mathrm{eq}}, \theta_{\mathrm{sr}}$ or θ, σ. It is worth noting that points (**a**) and (**d**) depend on the specific compliant actuators that the user intends to simulate. For this reason, we implemented these two blocks as *virtual* functions of the main C++ class that the users can customize according to the compliance models of their platform. More details regarding this are given in Section 3.1.3, while points (**b**) and (**c**) will be described in Section 3.1.1 and Section 3.1.2, respectively. In order to simulate different control schemes that can be physically realized on the real compliant-actuated joints, we defined several operation modes, as shown in Fig. 4 and Fig. 5. Furthermore, to increase the general applicability of the toolbox, we leave also the possibility of simulating a rigid joint. Then, according to the operation mode selected by the user, the plugin class computes the torque for the joint, simulated in Gazebo, in different ways. Analogously, the topics from which the plugin retrieves the input commands, have different names. More in details, referring to Fig. 4 and assuming that the simulated joint is named `joint_x`, the available operation modes and the relative command topics are given as follows:

[4] `https://www.ode.org/`

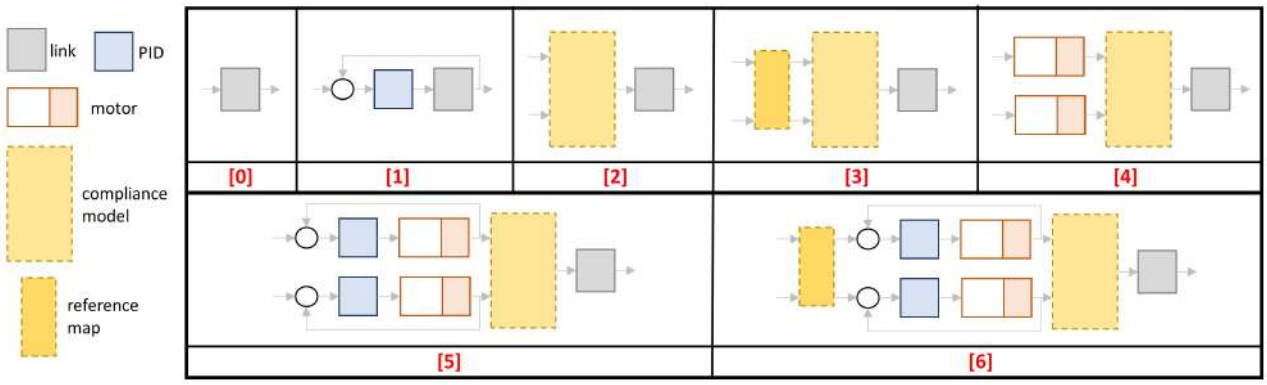

Figure 5. Control schemes of the operation modes implemented in the *CompliantActuatorPlugin*.

0) this mode allows controlling directly the link torque $\tau_{e,q}$ (rigid actuation), passed from the topic named `joint_x/torque_command`. The other topic (`joint_x/disabled`) is not used;
1) differently from the previous operation mode, this one exploits the PID controller to regulate the link position q to the desired one, i.e., $\hat{q}$, passed from the topic named `joint_x/link_command`. Even in this case, the other topic (`joint_x/disabled`) is not used;
2) in this mode the input references, used to evaluate the elastic torque $\tau_{e,\theta}$ or τ_{e,θ_i} from the compliance model, are passed through the topics `joint_x/reference_1` and `joint_x/reference_2`;
3) here the motor positions for the compliance model are retrieved from the reference map. Thus the topics refer to the equilibrium position and the stiffness preset, passed through the topics named `joint_x/equilibrium_position` and `joint_x/stiffness_preset`, respectively;
4) this mode exploits also the dynamics of the motors. Thus the input references consist of the desired motor torques and are passed through the following topics `joint_x/motor_1_command` and `joint_x/motor_2_command`;
5) similarly to mode (2), here the topics are named `joint_x/reference_1` and `joint_x/reference_2`, which, however, refer to the desired positions for the motors, regulated by the PID controllers;
6) as for the previous one, in this operation mode, the motor positions are regulated through the PID controllers, but the inputs refer again to the equilibrium position and stiffness preset variable, analogously to mode (3).

In addition to these topics, the plugin subscribes to a topic named `joint_x/external_torque` in which a possible external torque τ_{ext} can be defined. Furthermore, this plugin publishes the link position, velocity, stiffness, torque, and the value of the two references through the topic `joint_x/link_state`, while the state of the motors, where present, are published inside the topics named `joint_x/motor_1_state` and `joint_x/motor_2_state`. Finally, for the compliant-actuated joints cases, i.e., operation modes (2)-(6), the elastic torques are published as `joint_x/torque_el1_state` and `joint_x/torque_el2_state`.

3.1.1 DC motor dynamics block

In this block, referred to as *DCMotor* (orange box in Fig. 4), the dynamics of a simple dc motor has been implemented, in order to compute the motor positions that are required from the compliance model in operation modes (4), (5) and (6). We considered only the mechanical dynamics since the electric one is faster and, therefore, can be neglected. Thus, starting from (2), this block assumes the following dynamics

for each motor $s_j = \{1, 2\}$

$$\ddot{\theta}_{s_j} = \frac{1}{b}\left(u_{s_j} - d\dot{\theta}_{s_j}\right) , \tag{7}$$

where $b, d \in \mathbb{R}$ are respectively the inertia and the damping constants of the motor (reflected at the link) and defined in the plugin insertion (Fig. 6). The term $u_{s_j} \in \mathbb{R}$ includes the s_j-th component of the motor torque commands τ_{m}, the contribution of the elastic torques $\tau_{\text{e},\theta}$ in (2), and a simplified component for the torque friction derived from the Dahl model (Dahl, 1968) and implemented as in (Hayward and Armstrong, 2000), i.e., $\tau_{\text{f}} = K(\theta - w)$ in which w is evaluated with

$$w = \begin{cases} \theta + \frac{\hat{\tau}_{\text{f}}}{K}, & \text{if} \quad \theta - w < -\frac{\hat{\tau}_{\text{f}}}{K} \\ \theta - \frac{\hat{\tau}_{\text{f}}}{K}, & \text{if} \quad \theta - w > \frac{\hat{\tau}_{\text{f}}}{K} \end{cases} , \tag{8}$$

where K is the spring constant of the elastic model, while $\hat{\tau}_{\text{f}}$ is the amount of static friction torque to simulate on the joint. Despite its simplicity, this choice allows maintaining a passive implementation, independent from the sampling time adopted for the simulation. Hereinafter the subscript s_j will be omitted for the sake of clarity. To implement this dynamics in ROS, (7) must be reformulated in state-space form as follows

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -\frac{d}{b} & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} \frac{1}{b} \\ 0 \end{bmatrix} u , \tag{9}$$

where the state $x = [x_1,\ x_2]^T = [\dot{\theta},\ \theta]^T$. Then (9) should be discretized. Among the different methods (Quarteroni et al., 2010), we selected the forward Euler technique, for which $\dot{x}_i(k) = T^{-1}\left(x_i(k) - x_i(k-1)\right),\ i = 1, 2$, where $k \in \mathbb{Z}$ is the discrete-time variable and $T \in \mathbb{R}$ is the discretization time. This choice allows us to maintain a simple and efficient implementation. Nevertheless, other different approaches can be tested to improve the accuracy of the discretization. By applying the Euler integration method, (9) becomes

$$\begin{bmatrix} \dot{x}_1(k) \\ \dot{x}_2(k) \end{bmatrix} = \begin{bmatrix} 1 - \frac{d}{b}T & 0 \\ T & 0 \end{bmatrix} \begin{bmatrix} x_1(k-1) \\ x_2(k-1) \end{bmatrix} + \begin{bmatrix} \frac{T}{b} \\ 0 \end{bmatrix} u(k) . \tag{10}$$

3.1.2 PID controller block

In addition to the motor dynamics presented in (10), in the *CompliantActuatorPlugin* we also implemented the low-level position controller discussed in Section 2.2, used in operation modes (1), (5) and (6). This is done by implementing a simple PID controller of the following form

$$\bar{\tau}_{\text{m}} = k_{\text{p}} e + k_{\text{i}} \int e \text{d}t + k_{\text{d}} \dot{e} , \tag{11}$$

where $e = \hat{\theta} - \theta$ is the s_j-th motor position error with its first derivative $\dot{e}$, while $\hat{\theta}$ is the desired motor position. $k_{\text{p}}, k_{\text{i}}, k_{\text{d}} \in \mathbb{R}$ are the control gains of the PID, passed to the plugin as parameters. Equation (11) is evaluated at each step inside the controller block named *PIDController* (blue box in Fig. 4).

3.1.3 Customizable functions

In order to realize other custom plugins exploiting the proposed toolbox, the users must derive the main class presented in Section 3.1 and insert their compliance model, as well as passing the model parameters and define the reference map. This can be achieved by implementing the following *virtual* functions of the main class (dashed yellow boxes in Fig. 4).

- *tauElastic()*, this function is used to implement the compliance model, i.e., to compute the elastic torque components of the motors ($\tau_{\text{e},\theta}$ or τ_{e,θ_i}) as well as the elastic torque for the joint $\tau_{\text{e},q}$ and the stiffness value $\frac{\partial \tau_{\text{e},q}}{\partial q}$. The inputs required from this function are the motor θ_1, θ_2 and the link q positions;
- *eqPres2Refs()*, this function, instead, implements the change of variables, i.e., how the motor positions can be derived from the equilibrium and stiffness preset values. This depend on the type of compliant actuator to be simulated;
- *InitParams()*, this last function allows the user to create and define some useful optional tags for the plugin (see panel **B** in Fig. 6). For instance, referring to Fig. 6, it is possible to see that there are other tags inserted, besides the motors and controllers ones. These optional tags allow setting a_1, a_2, k_1, k_2, and the maximum deflection, which are parameters of the specific compliance model to be simulated, i.e., the qbMove Advanced actuator discussed in the following (shown in panel (iv) of Fig. 8).

3.2 Plugin Manager node

This part consists of a ROS node that communicates with the plugin presented in the previous section, as shown in Fig. 3. Given the modular structure of the blocks presented in Section 3.1, it results that a high number of topics are generated from the different joints. This fact, especially in the case of robots with a high number of joints, may increase the difficulty in passing the right commands to the relative joint and retrieving its information. For this reason, the goal of this node is to realize a more compact and clear implementation of the overall toolbox. To do so, according to Fig. 7, this node acts in two ways: i) allows passing the references to each joint starting from two simple and compact arrays; ii) subscribes to all the link and motor information published by the components of the plugin in order to collect them into arrays. To achieve the first objective, this node subscribes to two topics named `/reference_1` and `/reference_2`, that are no more relative to the single joint, but to the whole robot model, referred to as `robot_name`. In these two topics, there are arrays of the references, i.e., $\bar{\mathbf{r}}_1, \bar{\mathbf{r}}_2$, passed by the user. Then, according to parameters retrieved from a configuration file in which the joint and reference names are specified (see panel **A** in Fig. 6), it assigns these references to the relative joint. Depending on the type of compliant actuator simulated and the operational mode selected (see Fig. 4), the references will be different. More specifically we have that:

- for operation mode [0] the references is $r_{1,j} = \tau_{\text{e},q}$, while for mode [1] is $r_{1,j} = \hat{q}$, the desired link position. In these modes the second reference is not assigned;
- in mode [2] the references are $r_{1,j} = \theta_1$ and $r_{2,j} = \theta_2$, while in mode [5] the references are the desired one $r_{1,j} = \hat{\theta}_1$ and $r_{2,j} = \hat{\theta}_2$;
- in case of operation modes [3] the references are $r_{1,j} = \theta_{\text{eq}}$ and $r_{2,j} = \theta_{\text{sr}}$, and for mode [6], these are the desired ones, i.e., $r_{1,j} = \hat{\theta}_{\text{eq}}$ and $r_{2,j} = \hat{\theta}_{\text{sr}}$;
- while for the remaining mode, i.e., [4], the reference are to the motor torque $r_{1,j} = \tau_{\text{m},1}$ and $r_{2,j} = \tau_{\text{m},2}$;

The second objective of this node is achieved by subscribing to the following topics `joint_x/link_state`, `joint_x/motor_1_state`, and `joint_x/motor_2_state` of each joint that contain all the information relative to position, velocity, and torque of both link and motors, respectively, are published. Retrieved such information, the node copies those values inside three topics which deliver standard `JointState` messages type. These topics are named `/robot_state`, `/motor_1_state`, and `/motor_2_state` and again refer no more to the single joint but to the entire robot model.

Figure 6. Panel **A** shows an example of the configuration file required by the *Plugin Manager* node, while in panel **B** the template for inserting the plugin relative to the qbMove Advanced discussed in 3.1.3, is reported. It is worth noting that, besides required tags for the plugin, there are also optional tags that the users can define to modify the motor and controller parameters. Furthermore, there are customizable parameters that depend on the elastic functions implemented in the compliance model.

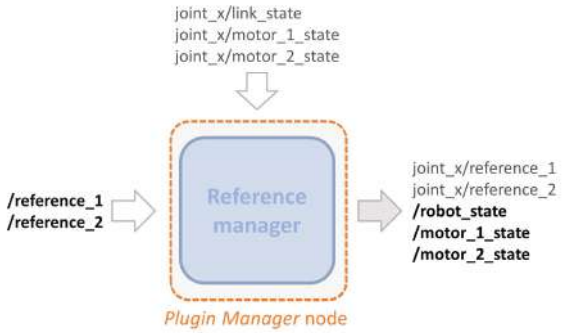

Figure 7. Detailed representation of the topics that the *Plugin Manager* node publishes and subscribes to.

3.3 Examples of derived plugins

To show the versatility of the toolbox, and to simulate different platforms, we made available the implementation of some compliant actuators presented by the VIACTORS consortium. More in detail, referring to Fig. 8, we implemented a simple SEA (e.g., the one presented in (Negrello et al., 2015)), the qbMove Advanced described in (Mengacci et al., 2021a), the Bidirectional Antagonistic Variable Stiffness prototype (BAVS) presented in (Petit et al., 2015), and the Actuator with Adjustable Stiffness II (AwAS-II) proposed in (Jafari et al., 2011). These derived plugins are released within the proposed toolbox and are named, *sea_plugin.cpp*, *qbmove_plugin.cpp*, *bavs_plugin.cpp*, and *awas_plugin.cpp*, respectively. The compliance models of these actuators can be found on the datasheet of each prototype, according to the standard specifications presented by Grioli et al. (2015). Then, the users can leverage on these examples, as well as on a template one made available within the toolbox, to realize other plugins according to their compliant platform.

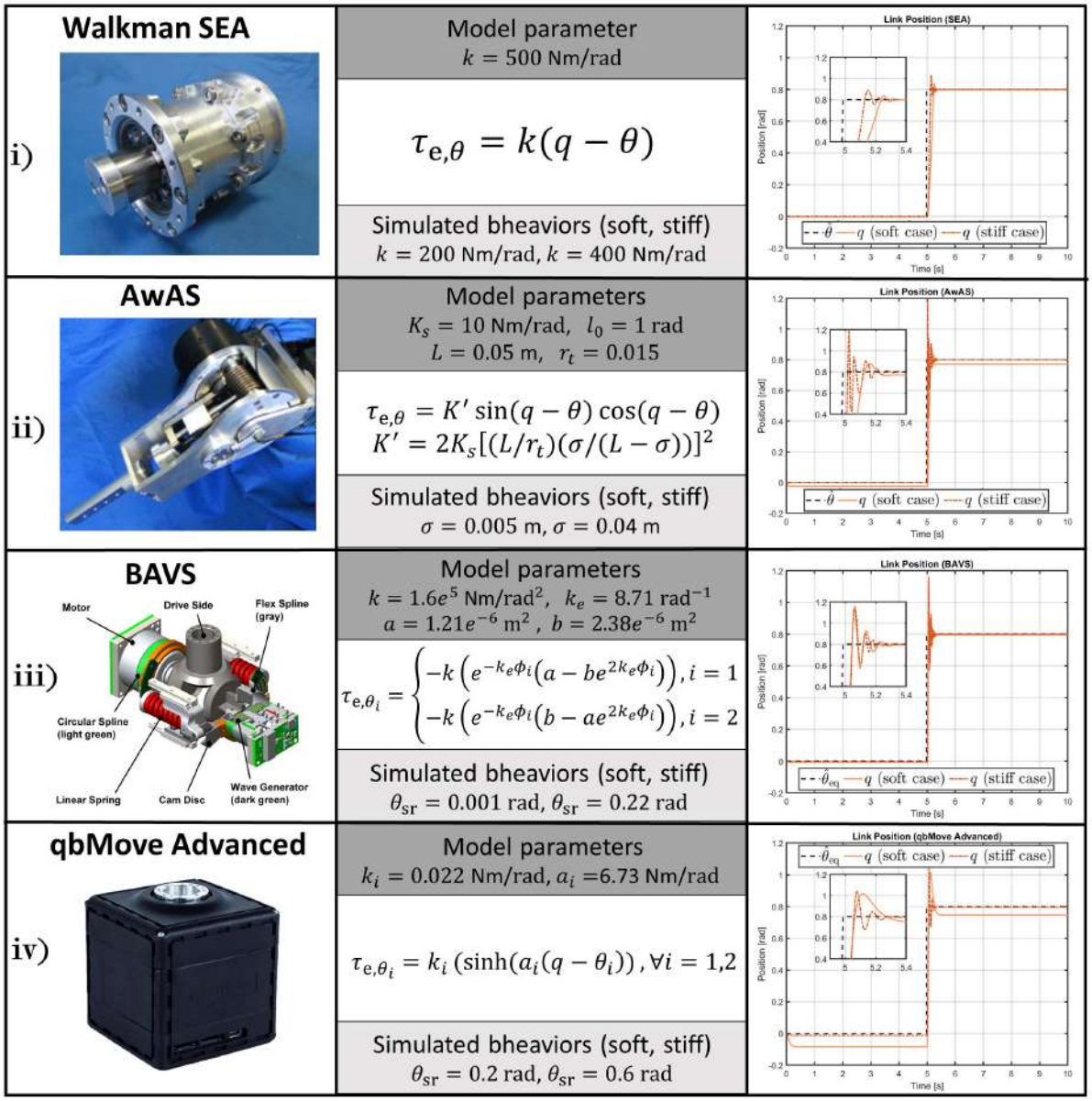

Figure 8. VIACTORS's compliance models implemented in the ROS-Gazebo toolbox and simulation results to the step response performed with different compliant behaviors (right-hand column). The 1-DOF structures simulate a mass of 0.2 kg attached to the output link, with the center of mass located at 0.04 m w.r.t. the joint axis, subjected to gravity.

4 TOOLBOX VALIDATION

To validate the performance of simulated ASRs realized with the proposed toolbox integrated with the Gazebo simulator, and to compare them w.r.t. the results obtained on the real-world platforms, we perform different simulation and experimental tests described in the following. More in detail, the first test reports the simulation of various compliance models, to show the capability of the proposed toolbox of simulating different compliant-actuated joints. Furthermore, a collection of experiments performed on real platforms[5] with multi-DOFs are presented to assess the reliability of the simulated counterparts, also in case of

[5] The URDF models of the simulated platforms, with the same mechanical characteristics of the real structures, are available in the GitHub repository of the toolbox.

potential interactions with the external environment. The latter test is also useful to validate how the proposed toolbox reproduces the compliant behavior of the real platform. The control scheme used for these validation tests is reported in Fig. 9. Starting from the encouraging validation results, then, we show how the proposed toolbox can reliably be used to transfer a control policy learned in simulation to the real-world platform, reducing the hardware time and leveraging the *Sim2Real* approach. Both simulations and experiments run on a desktop computer with a processor of Intel® Core™ i7-6700 CPU @ 3.40GHz×8, 16.5 GB RAM, equipped with Ubuntu 18.04.5 LTS, the ROS Melodic distribution, and Gazebo 9.

4.1 Simulated compliant actuator models

At first, we simulated the different compliant actuators discussed in Section 3.1.3. As shown in Fig. 8, we selected from VIACTORS's website four actuators: (i) Walkman SEA, (ii) AwAS II (VSA with Adjuster example), (iii) BAVS, and (iv) qbMove Advance (that are examples of VSA A-A case). Therefore, we considered four 1-DOF systems, each equipped with one of these compliant actuators and a mass of $0.2\,\text{kg}$ attached to the output link at $0.04\,\text{m}$ from the rotation axis of the joint, subjected to gravity. We analyzed the response of each system to a reference step of amplitude equal to $0.8\,\text{rad}$. Since we do not know the mechanical parameters of the motors mounted on the different compliant actuators, we choose as operation mode the number [3]. This allows us to control the joints directly through the compliance model inputs, according to the type of compliant joint. Therefore, an open-loop command is given as desired position $\hat{\theta}$ for (i) and (ii) and as desired link equilibrium position $\hat{\theta}_{\text{eq}}$ for (iii) and (iv). In addition to this, we considered two different compliant behaviors: *soft* and *stiff*. Referring to Fig. 8, the elastic parameters that we took to simulate the soft behavior are, $k = 200\,\text{Nm/rad}$ for the SEA actuator; $\sigma = 0.005\,\text{m}$ for the AwAS II; $\theta_{\text{sr}} = 0.001\,\text{rad}$ for the BAVS and $\theta_{\text{sr}} = 0.2\,\text{rad}$ for the qbMove Advanced, while for the stiff behavior, we considered $k = 400\,\text{Nm/rad}$, $\sigma = 0.04\,\text{m}$, $\theta_{\text{sr}} = 0.22\,\text{rad}$, $\theta_{\text{sr}} = 0.6\,\text{rad}$, respectively. The simulation results are reported in the right-hand column of Fig. 8. From these plots, we can conclude that the toolbox is able to simulate different compliance models effectively. Furthermore, from the simulations, we can notice how also the compliant behavior is reproduced. This is particularly noticeable from the (ii) and the (iv) case, for which the link regulation, in the case of soft behavior, results affected by the gravitational contribution. Conversely, where stiffening the output link, the regulation is performed correctly.

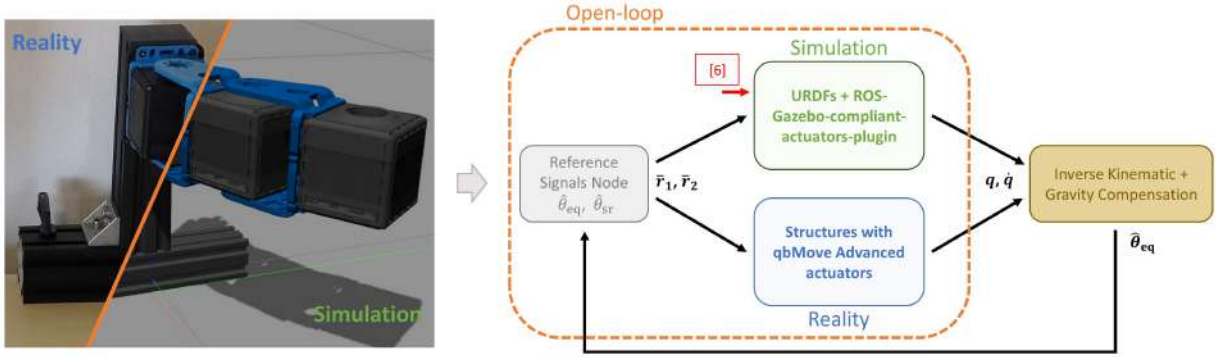

Figure 9. Control scheme used for both real experiment and simulated tests. Open-loop regulation (dashed orange box) has been exploited for the 1-DOF and 2-DOFs structures, while an inverse kinematic with gravity compensation approach has been used for the tests on the 5-DOFs arm.

4.2 Comparison of simulations and experiments

To assess the capability of the toolbox of replicating the performance of the real platforms in simulations, we carry out several experiments on different real and simulated platforms (Fig. 10): 1-DOF and 2-DOFs systems, and a 5-DOFs arm, both composed of qbMove Advanced VSAs. More in detail, the 1-DOF is used to investigate the frequency response of the simulated compliant-actuated joint. Instead, we tested on the 2-DOFs system the capability of the presented toolbox of reproducing the actuator behavior when a trajectory is commanded as desired link equilibrium position $\hat{\theta}_{\text{eq}}$, with a fixed stiffness preset $\hat{\theta}_{\text{sr}}$ (operation mode [6]). Furthermore, with the 5-DOFs robotic arm, we tested the stiffness reproduction when the system is interacting with the environment. Similar to the previous tests, for each experiment, we evaluated both the soft behavior and the stiff behavior, by changing the stiffness preset as $\theta_{\text{sr}} = 0.2\,\text{rad}$ and $\theta_{\text{sr}} = 0.6\,\text{rad}$, respectively. We also supposed damping and static friction at the joints, defined from the URDF file as $0.065\,\text{Nm/s}$ and $0.025\,\text{Nm}$, respectively. These tests are described in the following (see also the Video attachment).

4.2.1 Frequency response

The first comparison is performed on the 1-DOF platform, subjected to a chirp signal of $0.15\,\text{rad}$ amplitude and ranging from $0.5\,\text{Hz}$ to $6\,\text{Hz}$ of frequency. This test aims to evaluate the frequency response of the simulated compliant-actuated joint. As visible in Fig. 11, the behavior of the simulated platform is comparable to the real one. More in detail, the value of the peak deflection reached by the link output is similar between the two cases, either in the soft and in the stiff behavior. Furthermore, the link equilibrium position of the simulated case (red lines in Fig. 11) evolves as the real compliant actuator one. Despite this, however, there are few differences in the evolution of the output link and on the bandwidth of the

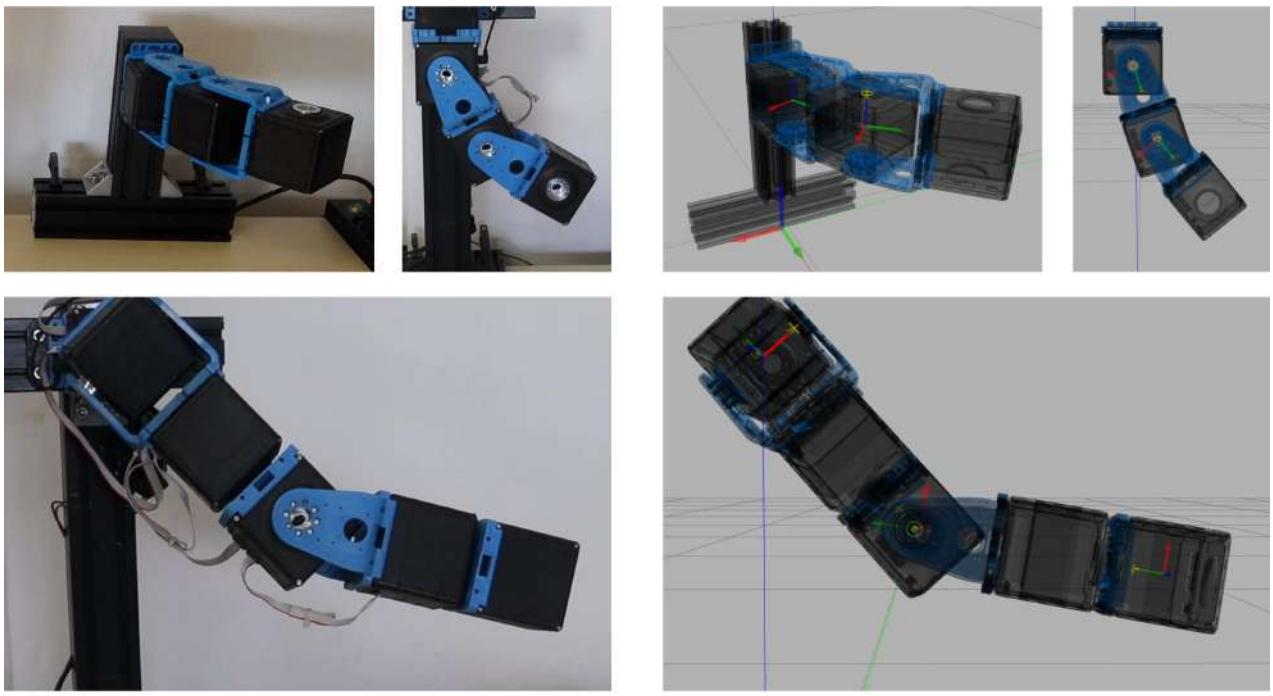

Figure 10. Structures used for the validation of the proposed ROS-Gazebo modules. The top figures in panel **A** and panel **B** show the 2-DOFs system in horizontal (left-hand side) and vertical (right-hand side) configuration, while the bottom figures depict the 5-DOFs experimental and simulated arm, respectively.

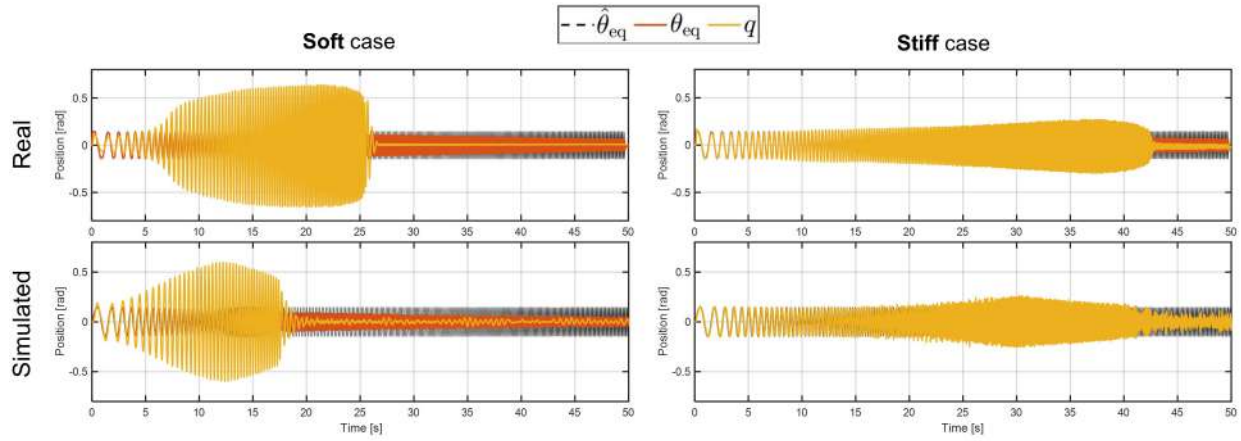

Figure 11. Experimental (top plots) and simulated (bottom plots) results of the frequency response test. The left-hand side shows the soft behavior, while the stiff behavior is reported on the right-hand side.

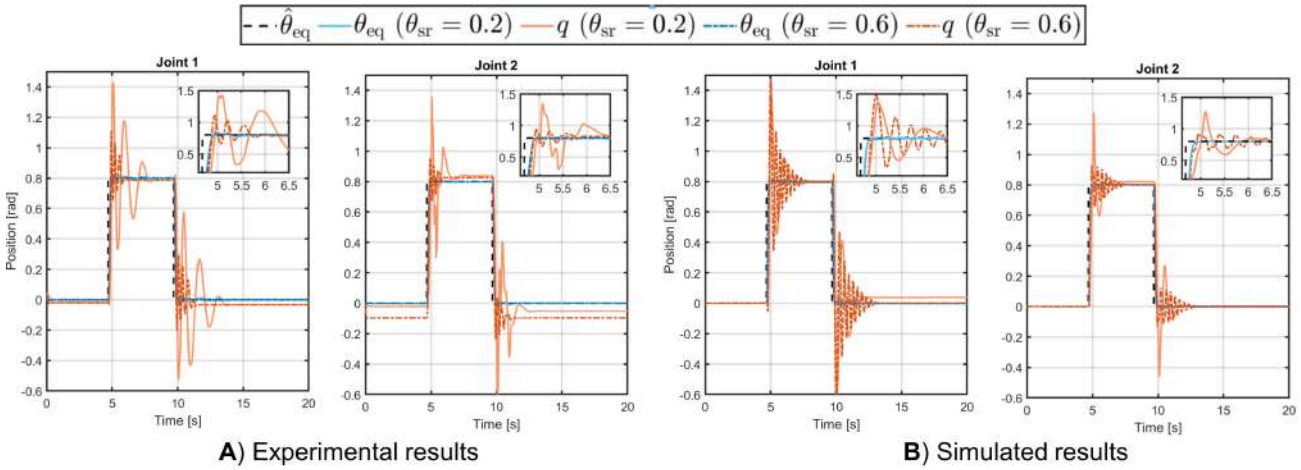

Figure 12. Experimental (panel **A**) and simulated (panel **B**) results of the 2-DOF structure in horizontal configuration following step references with soft (solid lines) and stiff (dashed-dotted lines) behavior.

two systems. These issues can be related to the joint parameters, i.e., the damping and the friction used by Gazebo to simulate the rigid-body dynamics, which have to be properly tuned in order to match the real performance. Moreover, non-linear unmodeled dynamics and simulation parameters, such as the sampling time, may affected these results. These facts will be better investigated in future works.

4.2.2 Step response and trajectory tracking

As stated above, to show the capability of the toolbox to reproduce the real actuator behavior when a $\hat{\theta}_{eq}$ is commanded, we used the 2-DOFs platform. Thus, we apply to each joint of this system, mounted in a horizontal configuration (top-left picture in panel **A** of Fig. 10), a step reference equal to the previous simulations. The experimental and simulated results are shown in panels **A** and **B** of Fig. 12. Referring to this figure, we can see that the simulated behavior is closely comparable to the real one. In particular, we can conclude that either the link equilibrium position (solid and dashed-dotted blue lines) and the link output (solid orange line) achieve the desired position. However, it is worth noting that the simulated case in stiff behavior presents high oscillation phenomena w.r.t. the real case. This is due to the fact that in the real-world platform, according to Zhang et al. (2021), a change of the stiffness preset affects also

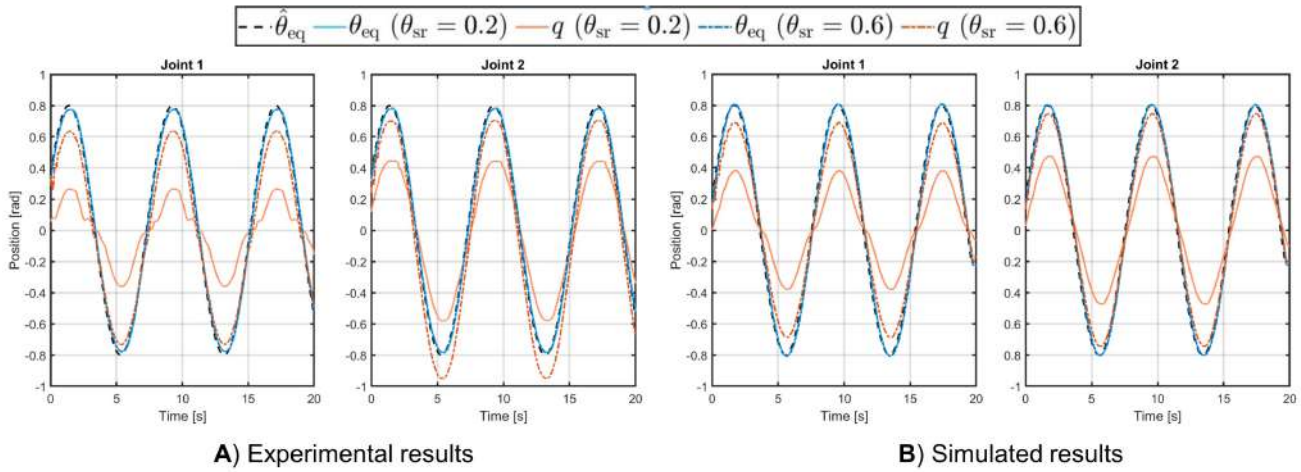

Figure 13. Experimental (panel **A**) and simulated (panel **B**) results of the 2-DOFs structure in vertical configuration performing a sinusoidal trajectory with soft (solid lines) and stiff (dashed-dotted lines) behavior.

non-linearly the damping seen at the link-side, while in the simulated case this parameter is kept fixed for both compliant behaviors. The second test conducted on this platform consists of a trajectory tracking to a sinusoidal reference. Then, the desired trajectory is computed in open-loop with the following equation

$$\hat{\theta}_{\text{eq}}(t) = H \sin(\omega t) \,, \tag{12}$$

where the amplitude is chosen as $H = 0.8\,\text{rad}$, while the frequency is set as $\omega = 0.8\,\text{rad/s}$. In this case, the system is mounted in a vertical configuration, as shown in the top-right picture in panel **A** of Fig. 10, thus it is subjected to gravity. From the results in Fig. 13 we can see that, differently from the previous test, the simulated behavior of the link in the soft case (solid orange line), is not equal to the desired one, despite the link equilibrium position tracks the reference (solid and dashed-dotted blue line in Fig. 13). This is due to the effect of the gravity torque on the compliance mechanism of the joint. Regarding the similarity of the simulated system w.r.t. the real platform, however, we can see that very similar behavior occurs also in the experimental platform, confirming that the toolbox is capable of reliably reproduce the real performance.

4.2.3 Stiffness reproduction during interaction tasks

In these experiments, we show the capability of the toolbox to reproduce the soft and stiff behavior of the system while interacting with the environment. In particular, we realized a 5-DOFs arm and commanded a desired trajectory to the end-effector. We implemented a first order closed-loop inverse kinematics (CLIK) (Siciliano et al., 2010) algorithm to find the corresponding desired link equilibrium positions $\hat{\theta}_{\text{eq}}$ for the different actuators. To compensate for the gravity effects on each joint, we computed and added to $\hat{\theta}_{\text{eq}}$ the deflection of the spring. Finally, we place a box of $2\,\text{kg}$ weight and dimensions $0.265 \times 0.265 \times 0.265\,\text{m}$ in the middle of the end-effector trajectory as an obstacle. For the simulated case, a wood-cardboard friction characteristics has been chosen for the contact between the box and the ground, by changing model parameters available in Gazebo[6]. As illustrated in Fig. 14, in the soft case, both in simulation and in the reality, the box stops the end-effector movement. Instead, in the stiff one, the arm moves the box to reach the desired end-effector position. These results, again, confirm that the compliant-actuated platform

[6] In particular, we imposed the <mu> tag of the surface friction equal to 0.23, which corresponds to the wood-cardboard friction coefficient available online.

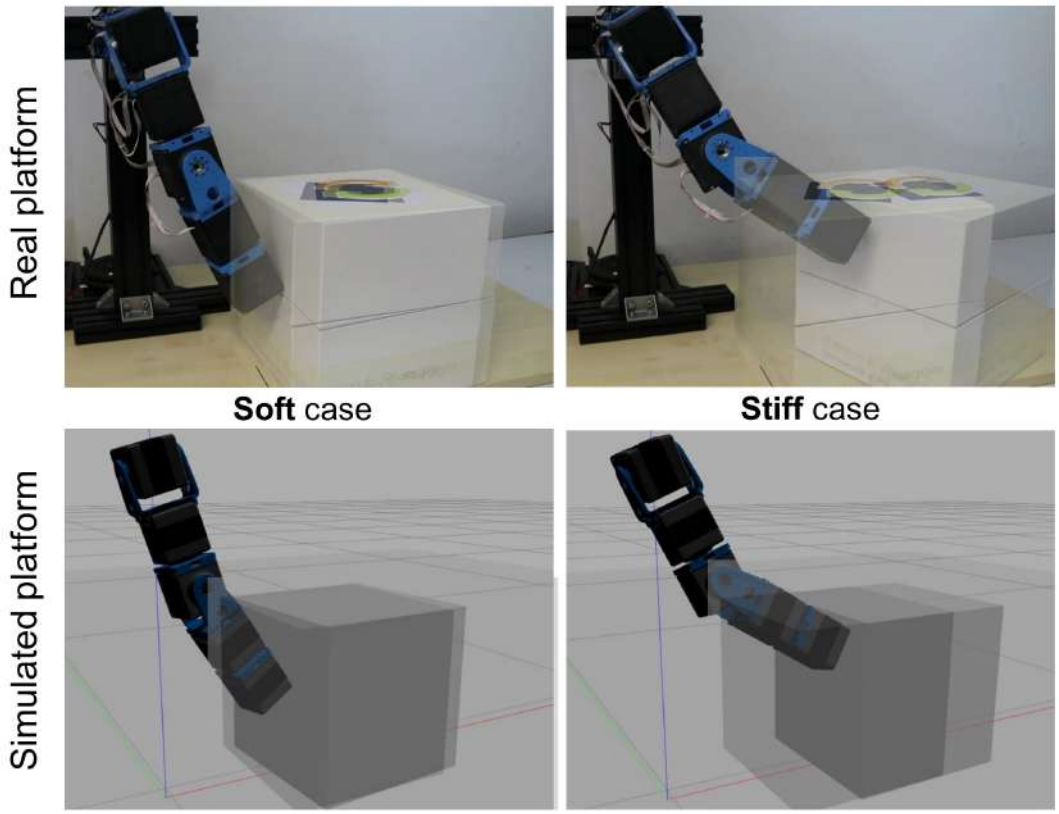

Figure 14. Experimental and simulated validations of the stiffness reproduction with the proposed ROS-Gazebo toolbox in case of an interaction task with a box of 2 kg weight. The right-hand side figures show the behavior in the soft scenario, while on the left-hand side the stiff case is reported. In the former case both the experimental and the simulated platforms are not capable to move the box, while if stiffened they can exert an increased force to the box able to move it forward.

simulated with the proposed toolbox is able of capturing the behavior of the real ASR even in the case of non-negligible interactions with the external environment.

4.3 *Sim2Real* example

To evaluate the real-world reliability of simulations executed with our toolbox, we performed a *Sim2Real* test in which a feed-forward control policy, learned iteratively in simulation, is used to drive a real-world structure. Iterative Learning Control (ILC) is a procedure through which one refines the time profile of the input that makes a robot execute a desired trajectory. We chose to test our simulations using the ILC paradigm for three different motivations. The first motivation is that ILC is recently being re-proposed as a promising technique to design the control of ASRs, and in particular those with variable stiffness (see, e.g. (Angelini et al., 2018)) since they do not alter the mechanical impedance characteristic of the ASR as feedback based controls do. The second motivation lies in the fact that ILC, being based on repeated iterations, highlights the usefulness of *Sim2Real* approaches in reducing the requirements in terms of hardware time. The third and final motivation is that the output of ILC is transferred to the hardware completely in feed-forward. This makes this test particularly challenging, since we avoid relying on the intrinsic robustness typical of feedback controller. We want to design the profile of the equilibrium position of the ASR shown in Fig. 10 (top-right picture) to follow a fifth-order minimum-jerk trajectory that goes from zero to $q_i = [0.8, 0.8]^T$ rad in 20 s. The ASR is a 2-DOF arm, subject to gravity, actuated by two

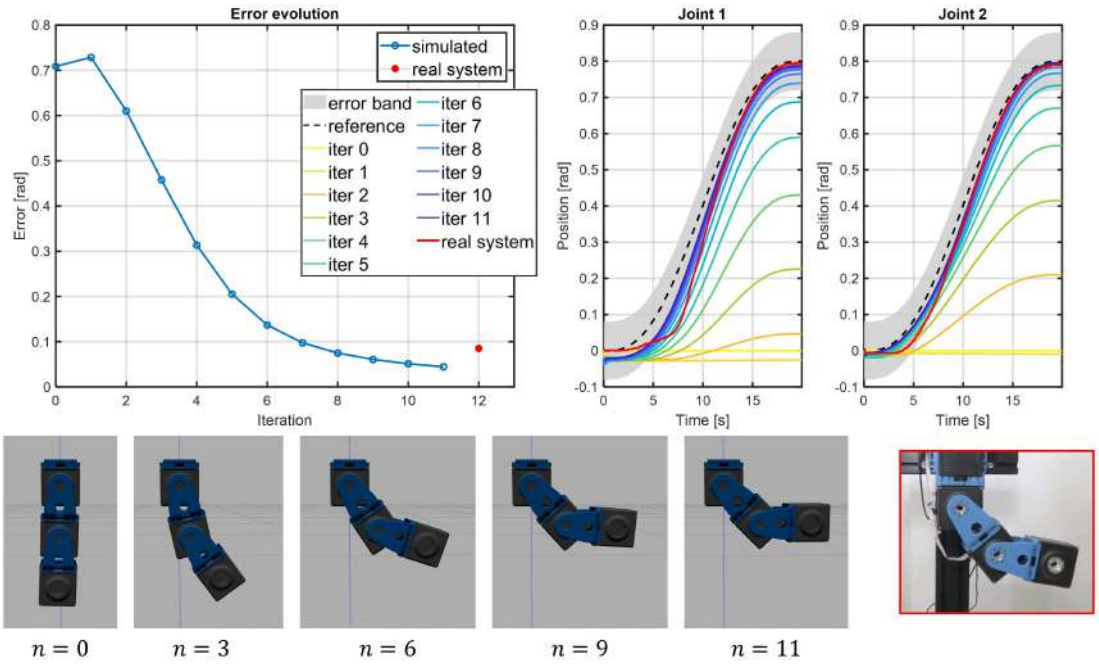

Figure 15. The Iterative Learning Control (ILC) procedure is executed on the simulated platform to retrieve the desired link equilibrium positions for the joints. Then, the control inputs are fed directly to the real system. Top-left plot shows the evolution of the error, computed as shown in Sec. 4.3, through the iterations, while top-right illustrates the trajectory tracking for each iterations and for the real test. The bottom figures show the frame sequences of the learning process and of the real hardware test.

qbMove Advanced actuators (panel **D** in Fig. 8). The parameters of the simulated compliance model were identified on the real platform using a procedure similar to that described in (Grioli et al., 2015). The resulting set is $k_1 = 6.257\,\text{Nm/rad}$, $a_1 = 0.01038\,\text{rad}$ and $k_2 = 3.9\,\text{Nm/rad}$, $a_2 = 0.08918\,\text{rad}$ for the first joint and $k_1 = 3.383\,\text{Nm/rad}$, $a_1 = 0.1015\,\text{rad}$ and $k_2 = 7.011\,\text{Nm/rad}$, $a_2 = 0.05944\,\text{rad}$ for the second joint. We set the value of the stiffness regulation parameter for both joints to $\theta_{\text{sr}} = 0.2\,\text{rad}$, which corresponds to the soft case considered in the previous experiments. Then, we apply an ILC scheme similar to that presented in (Angelini et al., 2018). According to this ILC scheme, at each iteration $n \in \mathbb{N}$ the link equilibrium position for the ASR is updated according to the value of the link equilibrium $\theta_{\text{eq}}(n-1)$ and the output link position error $e_q(n-1)$, computed at the iteration $n-1$. The update law is

$$\theta_{\text{eq}}(n) = \theta_{\text{eq}}(n-1) + K_{\text{off}} e_q(n-1)\,, \tag{13}$$

where K_{off} is the offline proportional gain, chosen iteration-dependent as $K_q e^{-(n/10)}$ with $K_q = 0.5$. We stop iterating once the maximum error on both joints is within a nominal tolerance band of $0.08\,\text{rad}$. The ILC algorithm converges after 11 iterations, after which the learned input is fed to the real hardware platform and executed.

The results of the ILC and of the hardware test are reported in Fig. 15. As visible in the top-left plot in figure, the iterations performed in simulations allow the system to learn the control input, i.e., the link equilibrium position, required to perform the trajectory tracking. After that, the same control input is used to drive the real system, achieving a comparable error. For both cases the error is computed as $e = ||\hat{q} - q||_2$. From the error plot, and from the trajectory tracking shown in the top-right corner, we can appreciate that the simulated and real platforms perform very similar. This can be also visually evaluated from the bottom plot in Fig. 15, where frame sequences of the ILC process and of the subsequent test on the real system

are shown (see also the Video attachment). Looking at the detail of the trajectories of the real robot, we can notice how the trajectory of the second joint is always within the nominal tolerance band of 0.08 rad, and is also very close to the simulation. For the first joint, on the other hand, we see how the nominal tolerance band is exceeded for a small time between 5 and 9 seconds. We believe that this slightly inferior performance is due to the combination of two effects. The first is the notorious difficulty in modeling static friction phenomena, which are hard to identify and tend to be time-varying. This difficulty affects, partially, our model and we aim at improving it in the future. The second motivation is that the first joint is affected by a larger load that changes with the angle of the second joint. Therefore, the small deviations from the nominal position of the second joint contribute to move the first joint out of its trajectory even more. Nevertheless, recalling also the purely feed-forward nature of the hardware experiment, we consider the performance to be more than acceptable. Finally, we remark that the results on the hardware are obtained without any iteration on the hardware side, thanks to the 11 iterations on the simulation, saving at least 220 seconds of hardware time. Therefore, we can conclude that the proposed toolbox can be a useful tool to reliably speed-up the design and control of ASRs based on a *Sim2Real* approach.

5 CONCLUSIONS

This work proposes an open-source ROS-Gazebo toolbox useful to simulate the dynamics of ASRs driven by compliant-actuated joints. The main goal of the toolbox is to allow the user to reliably simulate different dynamic models for the compliant joint, with or without considering the motor dynamics and a possible low-level controller. To achieve this, the toolbox leverages the ROS and Gazebo frameworks and is structured in two parts: one devoted to the implementation of a Gazebo plugin, which can be derived from the user, where the compliance models and the actuation dynamics are implemented, and the other one consists of a ROS node used to organize the input/output variables of the toolbox, intending to realize an easy-to-use interface. Furthermore, as an additional contribution, the users can customize the first part of the toolbox, implementing the compliance characteristics of their ASR. In this regard, to simplify this operation, together with the toolbox we released the implementation of different compliant actuators proposed by the VIACTORS's consortium. Simulations on these latter actuators, as well as several experiments on structures with multiple DOFs, equipped with VSAs at the joints, are carried out to validate the overall toolbox. These tests showed that the toolbox is able to effectively reproduce the performance of the real-world platforms and also to reliably simulate the compliant behavior of the ASR during interaction tasks with the environment. Motivated by these validation results, we also show how the toolbox, leveraging the *Sim2Real* approach, can be used to learn a control policy entirely in simulation, that can be then transferred directly to the real-world platform, maintaining satisfactory performance. Future works intend to improve the integration of the toolbox with existing ROS-Gazebo packages, e.g., the *ros_control* (Chitta et al., 2017), to realize a more general control toolbox also for the case of ASRs.

AUTHOR CONTRIBUTIONS

RM, GZ, and GG defined the structure of the proposed toolbox and contributed to its implementation. All authors contributed to the definition of the possible applications for the validation of the toolbox. RM and GZ wrote all sections of the manuscript. AB contributed expertise and advice. All authors equally participate to manuscript revision, reading, and approving the submitted version.

FUNDING

This project has received funding from the European Union's Horizon 2020 research and innovation program under agreement no. 101016970 (NI), no. 871352 (ReconCycle), no. 810346 (Natural BionicS), and no. 871237 (SOPHIA).

ACKNOWLEDGMENTS

The authors would like to thank Valeria Parnenzini, whose work inspired the development of this toolbox.

This is a preprint version of Mengacci et al. (2021b), published at `https://www.frontiersin.org/articles/10.3389/frobt.2021.713083/full`, DOI: `https://doi.org/10.3389/frobt.2021.713083`.

DATA AVAILABILITY STATEMENT

The open-source ROS-Gazebo toolbox presented in this article is part of the open-source Natural Machine Motion Initiative (NMMI, `https://www.naturalmachinemotioninitiative.com/`) and can be found at the following GitHub repository [**ROS-Gazebo-compliant-actuators-plugin**] [`https://github.com/NMMI/ROS-Gazebo-compliant-actuators-plugin`], together with the URDF models of the platforms used for the validation. The toolbox is released under the BSD 3-Clause license.

REFERENCES

Albu-Schäffer, A. and Bicchi, A. (2016). Actuators for soft robotics. In *Handbook of Robotics*, ed. 2nd (Springer), chap. 21. 508–513

Albu-Schaffer, A., Eiberger, O., Grebenstein, M., Haddadin, S., Ott, C., Wimbock, T., et al. (2008). Soft robotics. *IEEE Robotics & Automation Magazine* 15

Angelini, F., Della Santina, C., Garabini, M., Bianchi, M., Gasparri, G. M., Grioli, G., et al. (2018). Decentralized trajectory tracking control for soft robots interacting with the environment. *IEEE Transactions on Robotics* 34, 924–935

Cacace, J., Mimmo, N., and Marconi, L. (2020). A ros gazebo plugin to simulate arva sensors. In *2020 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE), 7233–7239

Chitta, S., Marder-Eppstein, E., Meeussen, W., Pradeep, V., Tsouroukdissian, A. R., Bohren, J., et al. (2017). ros_control: A generic and simple control framework for ros. *The Journal of Open Source Software* 2, 456–456

Collins, J., Chand, S., Vanderkop, A., and Howard, D. (2021). A review of physics simulators for robotic applications. *IEEE Access*

Dahl, P. R. (1968). *A solid friction model*. Tech. rep.

Della Santina, C., Catalano, M. G., and Bicchi, A. (2020). Soft robots. *Encyclopedia of Robotics*

Feldman, A. G. (1986). Once more on the equilibrium-point hypothesis (λ model) for motor control. *Journal of motor behavior* 18, 17–54

Grebenstein, M., Albu-Schäffer, A., Bahls, T., Chalon, M., Eiberger, O., Friedl, W., et al. (2011). The dlr hand arm system. In *2011 IEEE International Conference on Robotics and Automation* (IEEE), 3175–3182

Grioli, G., Wolf, S., Garabini, M., Catalano, M., Burdet, E., Caldwell, D., et al. (2015). Variable stiffness actuators: The user's point of view. *The International Journal of Robotics Research* 34, 727–743

Guizzo, E. and Ackerman, E. (2012). The rise of the robot worker. *IEEE Spectrum* 49, 34–41

Hayward, V. and Armstrong, B. (2000). A new computational model of friction applied to haptic rendering. In *Experimental Robotics VI* (Springer). 403–412

Ivaldi, S., Peters, J., Padois, V., and Nori, F. (2014). Tools for simulating humanoid robot dynamics: a survey based on user feedback. In *2014 IEEE-RAS International Conference on Humanoid Robots* (IEEE), 842–849

Jafari, A., Tsagarakis, N. G., and Caldwell, D. G. (2011). Awas-ii: A new actuator with adjustable stiffness based on the novel principle of adaptable pivot point and variable lever ratio. In *2011 IEEE International Conference on Robotics and Automation* (IEEE), 4638–4643

Kameduła, M., Kashiri, N., Caldwell, D. G., and Tsagarakis, N. G. (2016). A compliant actuation dynamics gazebo-ros plugin for effective simulation of soft robotics systems: Application to centauro robot. In *International Conference on Informatics in Control, Automation and Robotics* (SCITEPRESS), vol. 3, 485–491

Lentini, G., Settimi, A., Caporale, D., Garabini, M., Grioli, G., Pallottino, L., et al. (2019). Alter-ego: a mobile robot with a functionally anthropomorphic upper body designed for physical interaction. *IEEE Robotics & Automation Magazine* 26, 94–107

Mengacci, R., Angelini, F., Catalano, M. G., Grioli, G., Bicchi, A., and Garabini, M. (2020). On the motion/stiffness decoupling property of articulated soft robots with application to model-free torque iterative learning control. *The International Journal of Robotics Research* , 0278364920943275

Mengacci, R., Garabini, M., Grioli, G., Catalano, M., and Bicchi, A. (2021a). Overcoming the torque/stiffness range tradeoff in antagonistic variable stiffness actuators. *IEEE/ASME Transactions on Mechatronics*

Mengacci, R., Zambella, G., Grioli, G., Caporale, D., Catalano, M. G., and Bicchi, A. (2021b). An open-source ros-gazebo toolbox for simulating robots with compliant actuators. *Frontiers in Robotics and AI* 8, 713083

Negrello, F., Garabini, M., Catalano, M. G., Malzahn, J., Caldwell, D. G., Bicchi, A., et al. (2015). A modular compliant actuator for emerging high performance and fall-resilient humanoids. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)* (IEEE), 414–420

Petit, F., Friedl, W., Höppner, H., and Grebenstein, M. (2015). Analysis and synthesis of the bidirectional antagonistic variable stiffness mechanism. *IEEE/ASME Transactions on Mechatronics* 20, 684–695

Pratt, G. A. and Williamson, M. M. (1995). Series elastic actuators. In *Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on* (IEEE), vol. 1, 399–406

Quarteroni, A., Sacco, R., and Saleri, F. (2010). *Numerical mathematics*, vol. 37 (Springer Science & Business Media)

Siciliano, B., Sciavicco, L., Villani, L., and Oriolo, G. (2010). *Robotics: modelling, planning and control* (Springer Science & Business Media)

Spong, M. W. (1998). Underactuated mechanical systems. In *Control problems in robotics and automation* (Springer). 135–150

Tutsoy, O., Erol Barkana, D., and Colak, S. (2017). Learning to balance an nao robot using reinforcement learning with symbolic inverse kinematic. *Transactions of the Institute of Measurement and Control* 39, 1735–1748

Vanderborght, B., Albu-Schäffer, A., Bicchi, A., Burdet, E., Caldwell, D. G., Carloni, R., et al. (2013). Variable impedance actuators: A review. *Robotics and autonomous systems* 61, 1601–1614

Zambella, G., Lentini, G., Garabini, M., Grioli, G., Catalano, M. G., Palleschi, A., et al. (2019). Dynamic whole-body control of unstable wheeled humanoid robots. *IEEE Robotics and Automation Letters* 4,

3489–3496

Zhang, Y., He, L., and Wu, C. (2021). The effect of preload force on damping in tendon-driven manipulator. *Industrial Robot: the international journal of robotics research and application*