A Rapid Control Prototyping and Hardware-in-the Loop Approach for Upper Limb Robotic Exoskeletons Control

Giulia Bodo *, Federico Tessari, Stefano Buccelli, Matteo Laffranchi

Posted Date: 13 February 2024

doi: 10.20944/preprints202402.0744.v1

Keywords: Rehabilitative Robotics; Compliant actuation; Robotic Control; Upper Limb Exoskeleton; Rapid Control Prototyping

 doi:10.20944/preprints202402.0744.v1

Article

A Rapid Control Prototyping and Hardware-in-the Loop Approach for Upper Limb Robotic Exoskeletons Control

Giulia Bodo [1,2,*], Federico Tessari [3], Stefano Buccelli [2] and Matteo Laffranchi [2]

[1] PhD student in Mechanical Engineering at Politecnico di Torino, Torino, Corso Duca degli Abruzzi 24, 10129, Italy
[2] Rehab Technologies Laboratory at Istituto Italiano di Tecnologia, Genova, Via Morego 30, 16163, Italy
[3] Postdoctoral Associate, Mechanical-Mechatronic Engineer The Eric P. and Evelyn E. Newman Laboratory for Biomechanics and Human Rehabilitation Department of Mechanical Engineering Massachusetts Institute of Technology
* Correspondence: giulia.bodo@iit.it

Abstract: In the last decade, robotic-mediated rehabilitation has emerged as a potential solution to improve repetitive task training. Each device in this field has a unique development history shaped by engineers' expertise in specific programming languages or platforms. In this work we adopt an approach that tries to abstract from the final implementation with the aim to make control logic more shareable and understandable. The authors will present the outcomes of the application of a Rapid Control Prototyping strategy to an upper-limb robotic exoskeleton. A model-based design approach implemented on a real-time target machine is presented. This modern design approach was explored with several control strategies and was used to test the exoskeleton's performances. The proposed method highlights how it is possible to develop the entire control architecture in a single programming environment.

Keywords: rehabilitative robotics; compliant actuation; robotic control; upper limb exoskeleton; rapid control prototyping

1. Introduction

In an ever-evolving technological landscape, the development of new technologies demands the availability of rapid and efficient testing procedures. These procedures play a crucial role in expediting the validation process, ensuring the timely introduction of developed technologies to the market. This is essential to prevent the risk of technological obsolescence, which may occur if the testing procedures linger unnecessarily. In the context of rehabilitative robotics, over the past decade, several exoskeletons for upper limb rehabilitation were proposed (Float [1], Alex [2], Armeo [3], CleverArm [4] AnyExo [5], Harmony [6], Aramis[7]). Despite their significant differences, these devices share a common technological foundation: design a device that can be coupled with the human body, closely resembling the physiological movements of the shoulder and scapular complexes. Some unconventional implementation attempts have been presented so far, such as the Limpact [8] hydraulic exoskeleton, which remained in a stage of prototype development. Most of the rehabilitation devices on the market (Armeo Power [3], Harmony [6]), or in an advanced stage of development (Anyexo [5], Float [1]), are equipped with brushless DC motors, harmonic drive transmission and SEAs (Series Elastic Actuators).

Several of the aforementioned devices followed a Rapid Control Prototyping (RCP) approach in their early stages of development. Literature offers a variety of platforms for the control design: ARMin [9] runs on MATLAB/Simulink XPC target (now known as Simulink Real-Time), ANYexo runs on a control PC running a Linux operating system and interfaces with the hardware by EtherCAT communication; Harmony is operated by a real-time control system running Linux patched with RT-Preempt and communicates via EtherCAT. Furthermore, these exoskeletons share a similar system

 doi:10.20944/preprints202402.0744.v1

architecture, with a central control unit overseeing the coordinated actions of various motor control boards distributed throughout the robotic structure. In terms of control strategies, a remarkable consistency can be observed. These systems offer a range of control modes, from precise position control for passive mobilizations to more sophisticated and adaptable approaches like impedance and admittance control, commonly used in rehabilitative robotics to guarantee a compliant human-machine interaction [10]. In the development process of rehabilitative devices, RCP and Hardware In the Loop (HIL) emerge as pivotal approaches, allowing the quick implementation and testing of complex control strategies.

RCP approach enables early testing on hardware, even in a prototype stage, allowing for a swift assessment of how the control algorithms perform in a real-world setting. Additionally, RCP facilitates code reusability across different testing platforms, leveraging flexibility in hardware integration. The real-time application allows quick modifications to the control architecture.

By combining HIL and RCP, the developed software can be deployed and tested in a realistic environment. This approach allows for an early validation of both hardware and software characteristics, encompassing the testing of potential faults. The synergy between HIL and RCP supports parallel development of hardware and software, with continuous updates and testing. This concurrent approach increases the number of design and test iterations, expediting the definition of final device characteristics and, in turn, reducing time consumption and overall costs.

Several platforms are available for RCP and HIL applications. Among the commercial development environments there are: (i) dSPACE RCP by dSPACE [11], which exploits hardware and software tools to expedite the prototyping, testing, and iteration of control strategies in diverse domains; (ii) MathWorks, through MATLAB/Simulink Real-Time, enables real-time execution of Simulink models on dedicated hardware like Speedgoat's real-time target machines [12], making it ideal for comprehensive model-based control testing; (iii) National Instruments (NI) VeriStand [13] provides a software solution for configuring real-time testing applications. Additionally; (iv) Opal-RT (RT-LAB) [14], ETAS (LABCAR) [15], and Quanser [16] offer real-time simulation platforms tailored for both HIL and RCP applications, serving as crucial tools for testing embedded control systems.

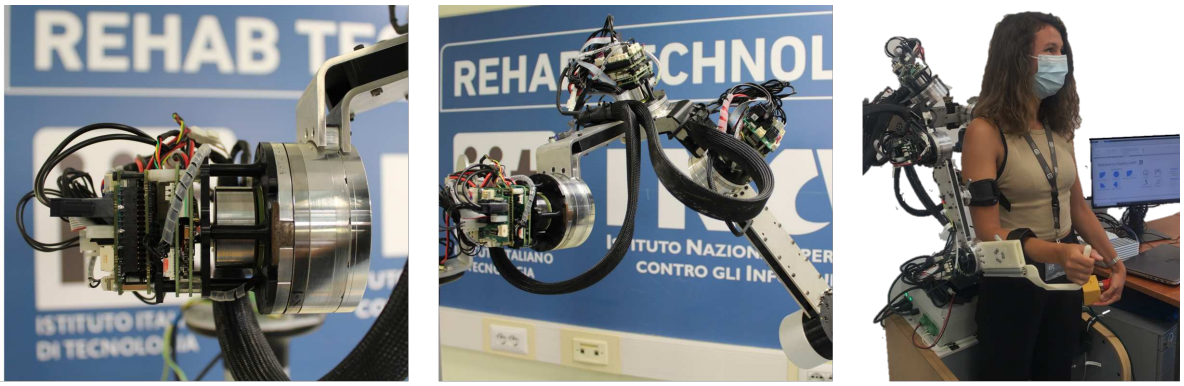
In this work, authors propose a RCP-HIL methodology targeted at rehabilitative robotics applications. The testing of the control architecture performances for an upper-limb exoskeleton prototype are presented. The peculiarity of the proposed method lies in the early adoption of a unified development environment and a single programming language. In the context of this project, the authors exploited MATLAB tools by MathWorks (Natick, MA, US). However, the same methodology can be leveraged in any other RCP development environment.

2. Hardware Platform for Rcp-Hil Test

2.1. Exokeleton Prototype

The hardware platform is a 6-Degrees-of-Freedom (DoFs) prototype of an upper limb exoskeleton (Figures 1 and 2). The prototype consists of two joints for the scapular complex (J1, J2), three joints for the shoulder complex (J3, J4, J5), and one joint for the elbow (J6). The mechatronics includes Series Elastic Actuators (SEA [17]) and motor control driver boards (EPOS4 Compact 50/8 (Maxon Group)). The control boards, mounted directly on the joints, are then connected, as well as the SEA sensors, via CAN (Controller Area Network) to the central control unit (Speedgoat Baseline real-time target machine [18]). This is further connected to a panel PC where the control interface is displayed.

 doi:10.20944/preprints202402.0744.v1



(a) (b) (c)

Figure 1. In the figure: (a) depicts the setup with EPOS boards for controlling the SEA joints and for the connection via CAN to the real-time target machine Speedgoat; (b) provides a close-up view of the spherical joint of the prototype used for real-time testing of developed control algorithms; (c) illustrates an example of the usability of the 6DoF prototype connected to Speedgoat, with control facilitated through a PC equipped with a GUI interface.

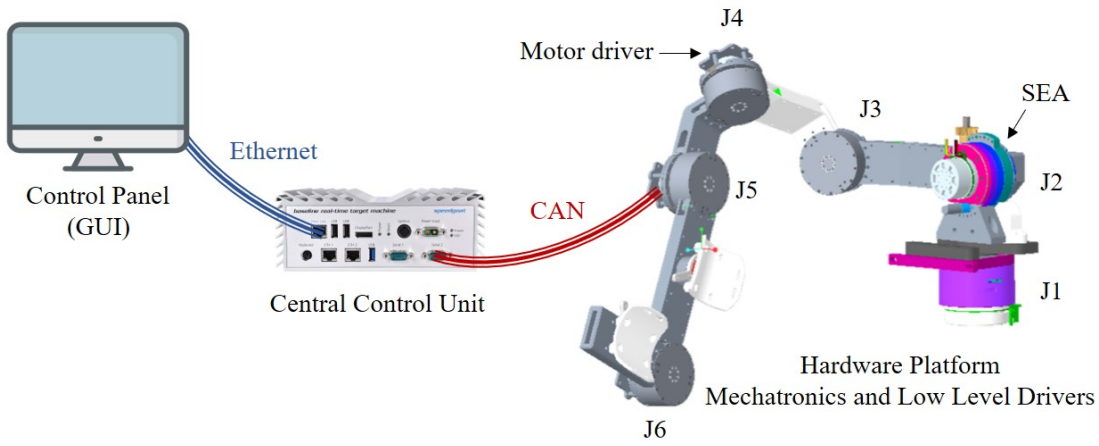


Figure 2. Hardware setup: EPOS motor controller and SEA sensors are connected via CAN to Speedgoat Baseline real time target machine. Speedgoat can communicate via ethernet with the host computer (Control Panel) receiving inputs from the GUI and providing feedback data about the running simulation.

2.2. Central Control Unit: Real Time Target Machine for RCP Development

To facilitate real-time simulations, a Real-Time Target machine has been employed, with Speedgoat serving as the platform for all conducted test. Speedgoat offers high performances thanks to multi-CPU target computers. It can be adopted in different areas like RCP and HIL systems. It is specifically designed to integrate with Simulink and Simulink Real-Time tools from MathWorks. This machine enables the execution of real-time applications developed within the MATLAB Simulink environment. The incorporation of Speedgoat Simulink driver blocks within the model streamlines the process, allowing for the automatic generation of real-time applications. Subsequently, these applications are downloaded and executed on the machine with ease. The target machine has integrated communication ports that enable the device to interact with the outside world. Speedgoat can communicate with other devices trough external ports, for the purpose of our experiments the CAN and Ethernet ports have been used, but many others communication protocols are available meeting the designer preferences (EtherCAT, PWM, SPI, I2C, SENT and more [19]).

 doi:10.20944/preprints202402.0744.v1

3. Control Design Methodology

The control strategy involves implementing all control levels (low, medium, high levels) within the same environment (MATLAB R2021b). Thanks to the use of various tools, it is possible to obtain a unique control model that includes setting up control drivers, acquiring and processing data from the mechatronic structure, and designing control algorithms (implemented following state machine logic). The steps of the proposed methodology are described below.

3.1. Geometric Model Description

The robotic kinematic chain can be described using the Denavit-Hartenberg convention (DH) or the modified Denavit-Hartenberg (mDH) convention [20]. For each frame (corresponding to a joint), it is possible to assign dynamic properties such as mass, center of mass, and the inertia tensor. To compute these properties, a CAD model should be utilized to facilitate the calculation of inertial properties with respect to the different joint reference frames. Subsequently, a rigid body tree (RBT) model (a geometric model characterizing both the kinematics and inertial properties of the robot) can be designed. The authors propose, as an example, the adoption of CREO PTC [21] as CAD software to obtain the dynamic characteristics and MATLAB Robotic Toolbox to build the exoskeleton rigid body tree model, as shown in Figure 7a. This RBT model is imported into the Simulink model and utilized to real-time compute the direct and inverse kinematics and the gravity torques to be compensated at each joint. The aforementioned procedure is described in Figure 3.

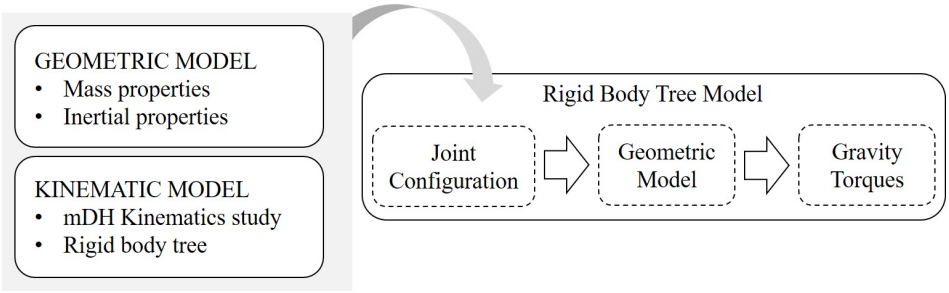


Figure 3. Workflow to develop the gravity torque estimation model: the dynamic model characterization for real-time gravity torque computation involves the utilization of various tools. In the CAD model of the prototype, the set of reference frames is redefined following the modified Denavit-Hartenberg (mDH) convention. Properties such as mass, center of mass, and inertia tensor for the different components of the device are then computed with respect to these reference frames. Using these dynamically derived properties, a rigid body tree model is constructed in MATLAB, adhering to the aforementioned mDH convention. This characterized model can be imported into the Gravity Torque function, which, in real-time, provides the joint torques necessary to maintain the robot at a specific configuration.

3.1.1. Design of the Low Level Control

The setup of the EPOS boards' controllers is carried out with the following steps: (i) communication of the motor specifications (torque gain, resistance, etc), (ii) communication of the PID gains for the position, velocity and current loops (that operates at $f = 1kHz$), (iii) communication of the desired operational limits. The communication protocol used to work with EPOS4 is the CANopen protocol. Inside the CANopen network the EPOS4 controllers are controlled as child nodes and coordinated by the Simulink model running on Speedgoat real time target machine. CAN protocol has been setup by employing IO614 reading and writing modules on Speedgoat Baseline Real-Time Target Machine [18]. With reference to the top layer of the model (Figure 4), *CAN SETTINGS* module is built to set up CAN connection exploiting CANopen protocol; *INPUT RECEIVED SIGNALS* module manages the input signals from SEA sensors and EPOS boards (acquired at $f = 1kHz$); *CAN SETTINGS TO*

 doi:10.20944/preprints202402.0744.v1

OUTPUT COMMANDS module sets up all the output control signals (computed and provided at $f = 1kHz$) to communicate via CAN with the sensors and EPOS motor driver boards.

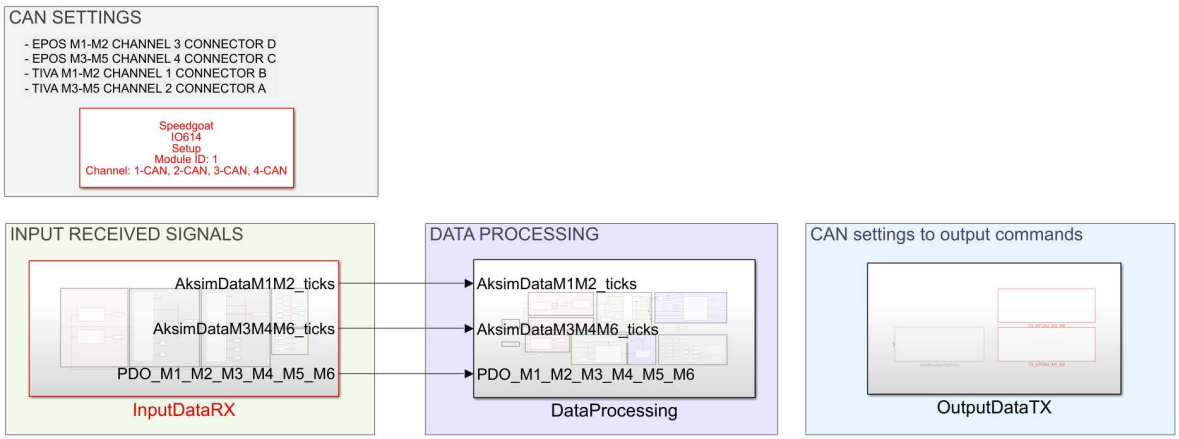


Figure 4. Top Layer of the Simulink control model. The model is built and then deployed on Speedgoat real time target machine. In this layer it is possible to identify four main blocks: CAN communication setup, readings of sensor data, processing of all the received data (this unit establishes the exoskeleton behaviour) and the output block that send over the CAN line the command signals.

3.2. Design of the Middle Level Control

Exploiting Simulink and Stateflow enables the straightforward implementation of multiple state machines to characterize the operational modes of the exoskeleton and potential state transitions. With reference to Figure 4, the dedicated module is *DATA PROCESSING*, that is the core module where all the control is implemented by means of control algorithms and state machines. The processing unit describes all the possible states transitions between the different behaviours of the device. The tasks performed in the Data Processing subsystem are: (I) Estimation of the SEAs torques; (II) Setup and reaching of the homing position (resting position of the exoskeleton); (III) Managing of the different working modes (passive exercises, exercise with compliance achieved thanks to impedance control, transparency, learn and replay); (IV) Computation of the pose of the exoskeleton end-effector; (V) Computation of the scapular elevation coupling following the method suggested in [1] [1]; (VI) Trajectory generation; (VII) Computation of the gravity torques from the robot dynamic model;(VIII) Torque control target generation (friction compensation, impedance, transparency).

The different control strategies aim to provide a variable assistance to the patient depending on the impairment level of the shoulder. Authors decided to implement different working modes, going from the first to the last modality, the goal is to increase the user engagement and decrease the exoskeleton assistance.

Passive user mobilization

This working mode exploits the position control. The exoskeleton moves the human limb in a rigid way, without keeping into account the interaction forces. A target trajectory is provided for each joint. The possible exercises are: shoulder flexion-extension, shoulder abduction-adduction and elbow flexion-extension. By means of the graphic user interface (Figure 6) it is possible to select the desired number of repetitions, timing (exercise duration) and maximum target angle. It is also possible to replicate, in position control, a recorded exercise. The model has a dedicated Stateflow state machine that can be used to perform the learn and replay. By exploiting the GUI buttons, it is possible to start

[1] The scapular elevation is function of the shoulder flexion. Once the shoulder flexion overcomes 60°, refer to Figure 7a, J2 is coupled with J5 with a ratio 1:3.

 doi:10.20944/preprints202402.0744.v1

the exercise recording (the exoskeleton goes in transparency mode and can be moved freely). Then joint position data are recorded, stored and used as target trajectory for the replay.

Compliant exercises: Impedance control to provide modular assistance during rehabilitative exercises

Impedance control can be exploited to support the patient in performing voluntary movements. Impedance control manages the capacity to exchange force between the controlled system and the subject or the environment. Linear impedance parameters, like stiffness and viscosity, are described as second-rank twice covariant tensors [22]. This impedance embodies the correlation between force and motion dictated by the supervisor, encompassing both the static force/displacement relationship and any dynamic terms necessary for controlled dynamic behavior [22]. Essentially, it dictates whether the robot exhibits rigidity or compliance upon interaction with external objects or surfaces. It belongs to the category of partially assistive controls [10], aiming to not only provide physical support but also to maintain the subject's motivation, training intensity, confidence in using the affected limb, and prevent negative reinforcement [23]. A tunable support level can be provided depending on the residual motor functionality of the user. The assistance level can be selected on the GUI, the target trajectories can be the ones of predefined exercises or a recorded complex trajectory stored during the learn-and-replay procedure. Impedance control can be used to ensure safety during human-machine interaction: it allows to dissipate torques and forces at joint level and not to the external environment.

By tuning stiffness (k_v) and viscosity (β_v), it is possible to provide a modular assistance increasing the user engagement while decreasing the provided support. The adopted impedance algorithm is developed in the joint space.

$$\Delta q = q_{target} - q_{actual} \tag{1}$$

$$\tau_{interaction} = \tau_{SEA} - \tau_{gravity} \tag{2}$$

$$\tau_{target} = \Delta q \cdot k_v + \dot{q} \cdot \beta_v + \tau_{interaction} \tag{3}$$

For each joint the control scheme shown in Figure 5 is applied. Simulink allows to fully explore the potential of real time computation. With reference to Figure 5, the Joint target position is real time computed for the standard exercises trajectory as a linear interpolation between the start and end position, set by the user into the GUI. This has to be performed in the desired amount of time and repeated for the selected number of repetitions. The trajectory can also be recorded during a learn process. The joint actual position and speed are provided from EPOS board. For this implementation a reference velocity $\dot{q}_{tg} = 0$ has been considered at all times. The sensed actual torque is computed by the model starting from the SEA sensor encoders readings (measure of the SEA spring deformation $\Delta\theta$), as explained in [17] :

$$\tau_{SEA} = ((\theta_{out} - \theta_{in}) - \Delta\theta_{offset}) \cdot k + \Delta\tau_{lm} \tag{4}$$

respectively $\theta_{out}, \theta_{in}$ and $\Delta\theta_{offset}$ are: the angular displacements after and before the spring, the mounting encoder offset, $\Delta\tau_{lm}$ is the linear model intercept and k is the stiffness of the adopted compliant element. The gravity torque is obtained from a dedicated Simulink block of the robotic toolbox as explained in Figure 3. All the aforementioned information is compared and elaborated to generate the control signals for the motor boards.

 doi:10.20944/preprints202402.0744.v1

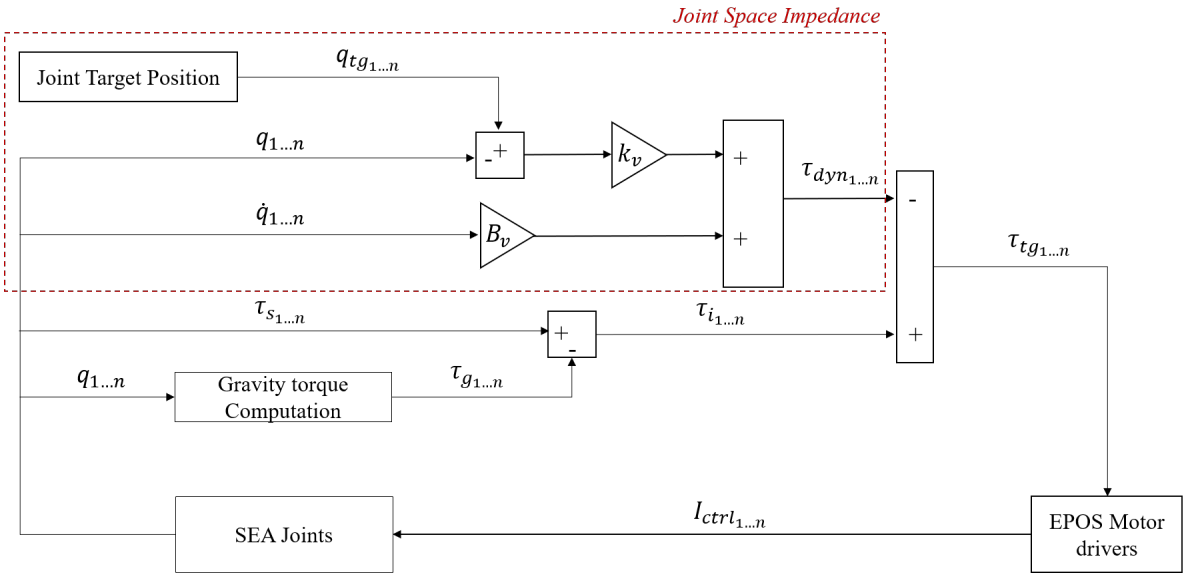


Figure 5. The impedance control scheme employs the presented algorithm individually for each joint to compute the control signal. The reference target, representing the desired position, has to be reached by ensuring a certain compliance between the user and the device. The difference between the target and actual positions represents the delta displacement of a virtual spring, that multiplied by a virtual stiffness coefficient (k_v) provides the spring torsional torque. The actual joint speed introduces a viscous behavior, determined by the viscous coefficient (β_v). The interaction torque, calculated as the difference between the estimated SEA torque and the gravity torque, serves as the feed-forward term. This ensures that the user can displace the exoskeleton in a compliant manner.

Algorithm 1 Friction compensation: the algorithm implemented in the dedicated state machine for friction compensation involves monitoring the rotational speed of the joint. Depending on the speed value, the compensation effect is determined, mirroring the behavior of a damper. Specifically, if the speed is below a minimum threshold, denoted as $\dot{q}_s$, it indicates that the joint is initiating movement. To overcome initial static friction, the compensation effects against friction are amplified. Within a predefined ideal speed range for device activities, the compensation contribution remains constant. However, if the speed exceeds a high threshold, denoted as $\dot{q}_f$, a gradual effect begins to compensate less against friction until minimal compensation is achieved at the predefined maximum speed, $\dot{q}_{max}$.

if $|\dot{q}| < \dot{q}_s$ **then**
  $\beta_{fr} \leftarrow \beta_s$
else
  if $|\dot{q}| \geq \dot{q}_s$ and $|\dot{q}| \leq \dot{q}_f$ **then**
    $\beta_{fr} \leftarrow \beta_f$
  end if
  if $|\dot{q}| > \dot{q}_f$ **then**
    $\beta_{fr} \leftarrow \beta_m \cdot (\dot{q}_{max}/|\dot{q}|)$
  end if
end if

Transparency: gravity and friction compensation

Transparency mode is implemented in order to let the user perform free movements, exploring the working space minimizing the exoskeleton weight and friction effects that can interfere with the smoothness of the performed movement. The motor control is performed in torque (current), the control signal is computed from the sum of different components: the gravity torque, the friction compensation torque and the interaction torque (obtained from the SEA sensed torque). This control exploits two different layers:

- Gravity compensation: it is needed to compensate the gravity effect to avoid that the structure collapses under its weight, thus a reliable gravity compensation model is needed. The obtained rigid body tree model (Figure 3) is exploited in Simulink by the Gravity Torque block of the

 doi:10.20944/preprints202402.0744.v1

Robotic Toolbox allowing to real-time compute, given a certain joints' configuration, the gravity torques to be compensated.

- Friction compensation: frictions can result in a lack of smoothness during the movement, especially at low speeds when static frictions need to be won. A Stateflow state machine has been implemented to evaluate, based on the actual motor speed, a compensatory friction torque used to overcome friction dissipation. The friction contribute is computed as $\tau_{fr} = -\beta_{fr} \cdot \dot{q}$, with β_{fr} parameter evaluated by the state machine in a dynamic way following Algorithm 1. β_{fr} is evaluated at each interaction. The highest compensation for overcoming frictions occurs at low speeds when static frictions need to be overcome. The coefficient β_f remains constant within an accepted speed range for movement in transparency. If the motor speed exceeds a specified threshold, indicating that the joint is moving too fast, β_f takes on a different value, increasing the viscous effect until reaching the maximum set velocity, thereby implying minimal support in overcoming frictions.

3.2.1. Design of the High Level Control

The deployed control model is integrated with a Graphic User Interface created using the MATLAB App Designer Tool. The GUI will serve as interface between the user, the control model and the real time target machine. Through this GUI, users can dynamically select and customize the exoskeleton's behavior, issuing commands (Figure 6) and monitoring key data for the prototype performance evaluation.

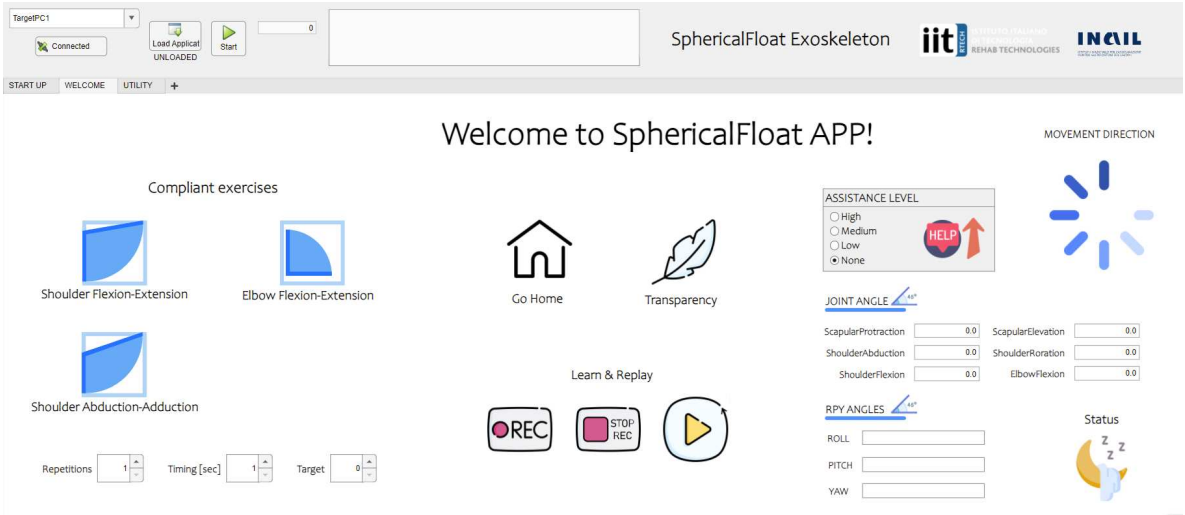


Figure 6. Graphical User Interface Welcome Panel: Within this tab of the GUI, crafted using the App Designer MATLAB Tool, users can establish the preferred working mode of the exoskeleton. This includes initiating straightforward exercises under passive or compliant control (with selectable levels of assistance), configuring the number of repetitions, timing, and target positions for the exercises. Users can command the robot to return to the Home position or activate transparency mode. Additionally, functionalities such as learning and replaying are accessible. The GUI provides comprehensive information about the current phase of the exercise, the direction of the movement to be followed, the status of the machine (sleep, operative, fault), also, information about the joints position and the end-effector orientation angles are avaiable.

4. Results

The 6Dof device prototype (shown in Figure 1) was tested in its various control modes to activate different states of the state machine. Specifically, the kinematic mode was tested with the execution of proposed exercises, the assistance mode with different levels of support provided, and the transparency mode aimed at using the device without the perception of friction and gravity forces.

 doi:10.20944/preprints202402.0744.v1

Table 1. In the table, the maximum errors and root mean square errors (RMSE) and absolute errors ($\epsilon_{\max}$) for the considered joints are computed by comparing the experimental measurements of SEA sensors with the output from the built-in gravity torque calculation function in the MATLAB Robotic Toolbox. The experiment conducted on the prototype involved the exoskeleton covering various zones within the workspace while operating in position control. This was carried out to validate the consistency of the SEA sensor readings with respect to the model characterized in MATLAB.

$JointID$	$RMSE[Nm]$	$\epsilon_{\max}[Nm]$
J3	0.2877	0.8338
J4	0.3101	0.8703
J5	0.0649	0.3449

4.1. HIL Testing: Validation of Gravity Compensation Model

Real-time experiments have been conducted to validate the dynamic model obtained form the characterization of the Rigid-Body-Tree shown in Figure 7a. Thanks to the Learn and Replay mode it was possible to record, save and reproduce a complex movement that covered the workspace. During the movement data relative to the gravity torques computed by the model and the sensed SEA torques were recorded. Then it was possible to compare the two measurements. In Figure 7b are shown the Gravity Torques computed from the dynamic model and recorded by SEAs for the spherical joint complex (J3, J4, J5). The computed RMSEs and absolute errors, between the SEA measure and the gravity compensation model are reported in Table 1.

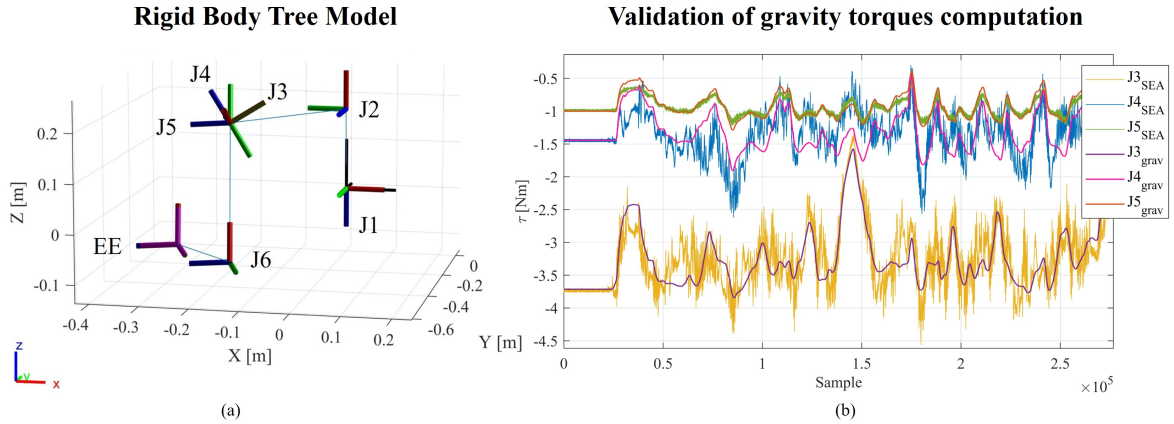


Figure 7. (a) Rigid body tree representing the kinematic chain of the 6-degree-of-freedom prototype. The Modified Denavit-Hartenberg (mDH) convention was followed for describing the kinematics. Each reference frame, representing a rotational degree of freedom, is associated with the dynamic properties of the body, including mass, center of mass coordinates, and the inertia tensor. (b) Graphical comparison between the torques estimated by the model and the ones measured by SEAs sensors. During the test the exoskeleton was moved in position control without a user or a load. In the plot are reported, for each joint, the torques estimated by the SEAs and the ones real-time computed from the model through the robotic toolbox.

4.2. HIL Testing: Compliant Modular Assistance

In the field of impedance control, a control has been developed with selectable levels of assistance (Table 2) accessible through a graphical interface. The Figure 8 illustrates test conducted at three different assistance levels during a shoulder flexion exercise, specifically focusing on joint J5, which plays a significant role in this movement. The graph in Figure 8 displays both the target trajectory and

 doi:10.20944/preprints202402.0744.v1

the actual trajectories executed by the user, exploiting the dedicated joint, throughout the exercise in assistive mode.

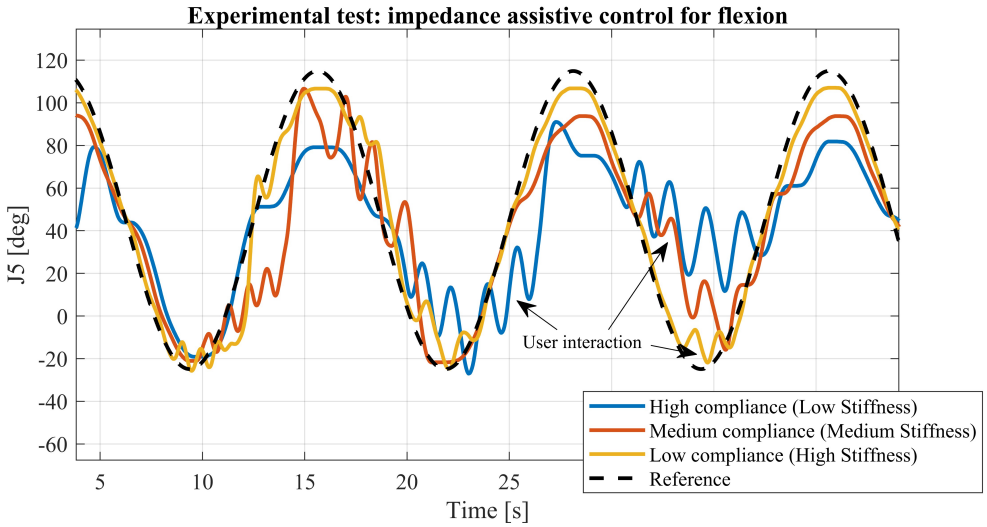


Figure 8. Experimental results of impedance-based assistive control: Three levels of assistance were tested by varying the stiffness-viscosity parameters of the impedance control (stiffness-viscosity couples are reported in Table 2). In the figure, the black line indicates the joint reference target, while the blue, orange and yellow lines represent the results of three test at different assistance levels. The displayed oscillations are the consequences of human interaction with the robot. It is noticeable that lower assistance (high compliance, low stiffness) allows for greater interaction with the robot, enabling the user to deviate significantly from the target trajectory. Higher assistance (low compliance, high stiffness) leads to better adherence to the target trajectory, albeit restricting interaction with the device to some extent.

Table 2. Pairs of stiffness and viscosity parameters (k_v and β_v) utilized in the impedance dynamic model to implement three different levels of assistance: decreasing k_v corresponds to lower support and higher compliance.

Compliance	$k_v[Nm/rad]$	$\beta_v[Nm \cdot s/rad]$
High	30	1.2
Medium	10	1.2
Low	5.5	0.6

5. Discussion

In this study, the authors introduced a methodology tailored for rapid control prototyping of control strategies in the field of rehabilitative robotics. Utilizing MATLAB-Simulink, the control architecture was exported and executed in real-time on the Speedgoat target machine: high-level code (Simulink) was automatically transformed into C/C++ (build duration: 0h 5m 15.516s) code on a real-time target machine. Various control strategies, with a focus on those incorporating compliant control techniques such as transparent and impedance control, were tested to provide adaptable assistance to users.

The authors presented the hardware platform used to test the 6-DoFs upper limb exoskeleton (Figures 1 and 2). The geometric model of the exoskeleton, obtained from CAD software, was utilized to design a rigid body tree model (Figure 7a) encompassing both kinematic and dynamic properties.

 doi:10.20944/preprints202402.0744.v1

Successively, the control model implemented in Simulink (Figure 4), explaining its state machine logic and various functionalities, is presented. Additionally, they showcased the graphical interface designed to facilitate device usage and testing (Figure 6).

Validation tests (Figure 7b) for SEA sensor readings and the dynamic model designed for gravity torque compensation yielded comparable results (see Table 1). The observed errors can be attributed to inertial and viscous effects not accounted for in the proposed dynamic model, which relies on MATLAB's Robotic Toolbox's built-in function. The proposed model (as depicted in Figure 3) only performs gravity compensation based on positions, neglecting inertial effects due to acceleration. However, within the context of rehabilitation, this compromise is acceptable given the typical low velocities and accelerations involved. Since the inertial properties of the system can be obtained from CAD, a future step could involve utilizing the rigid-body-tree to generate the mass matrix and also performing inertial component compensation.

Assistive control testing (Figure 8), utilizing impedance in joint space (Figure 5), was instrumental in selecting and testing various combinations of stiffness and viscosity. The objective was to achieve the tuning of modular assistance levels tailored to rehabilitation requirements.

Experiments were conducted with three levels of assistance, adjusting stiffness-viscosity parameters of impedance control (values listed in Table 2). Figure 8 illustrates the results, with the black line representing the target trajectory for the joint, while the blue, orange, and yellow lines depict outcomes of tests conducted at different assistance levels. The observed oscillations result from human interaction with the robot. By assigning a different spring-damper behavior, variable assistance could be selected. Lower assistance (high compliance and low stiffness) permits increased interaction with the robot, allowing users to deviate significantly from the target trajectory. Conversely, higher assistance (low compliance and high stiffness) ensures better adherence to the target trajectory, albeit with limited interaction with the device.

In this context, RCP facilitated various interactions to experimentally evaluate suitable pairs of k_v and β_v by dynamically adjusting parameters in real-time and observing the device's behavior.

While the proposed method offers simplicity and abstraction from low-level issues, facilitating rapid testing of control models, it has limitations. Specifically, its inability to access the underlying algorithms of the adopted functions and toolboxes. Despite this limitation, the authors believe that this method strikes a balance between ease of use and customization, prompting users to consider the trade-offs between simplicity, detailed control and flexibility.

6. Conclusions

Developing an exoskeleton is a complex endeavor that demands a multidisciplinary engineering effort. Activities spanning from mechanical design to electronic and control software design can be highly time-consuming, often impeding the development of more sophisticated control strategies. Therefore, especially in the initial iterative phases, having a fast and effective method to develop a first prototype is essential.

The proposed method aimed to expedite control prototyping and hardware testing by consolidating the entire control architecture into a single interface. The results underscore how this objective was achieved through the integration of system modeling, low-level driver configuration, control strategy development, and user interface implementation using the same language within a unified development system. This approach facilitates rapid implementation and testing, a challenging task to achieve using conventional development tools, while also obviating the need to switch between different programming languages.

In the proposed case, the authors use the MATLAB/Simulink environment, but this is only one of the possible options. Overall, this approach enables most control systems to undergo testing in the RCP phase before transitioning to the real machine.

The availability of a real-time machine can play a crucial role in ensuring a high standard and efficiency in developing cutting-edge technologies.

 doi:10.20944/preprints202402.0744.v1

7. Open Source Repository

An open-source repository is available with MATLAB-Simulink codes, models, and the graphical user interface used for the presented study at [https://github.com/giuliabodoiit/RCPmodelForExoskeletonControl_Bodo].

Acknowledgments: This work was funded by the Istituto Nazionale per l'Assicurazione contro gli Infortuni sul Lavoro (INAIL) under grant agreements "PR19-RR-P2 - RoboGYM" and "PR23-RR-P2 - ClinicEXO". Acknowledgment is owed to the RoboGym team within the Rehab Technologies Lab for their invaluable assistance in both the development and maintenance of the prototype throughout the testing phase. Special appreciation is extended to Luca De Guglielmo and Gianluca Capitta for their invaluable contributions in designing the prototype and setting up the hardware platform.

References

1. Buccelli, S.; Tessari, F.; Fausto, F.; Guglielmo, L.D.; et Al., G.C. A Gravity-Compensated Upper-Limb Exoskeleton for Functional Rehabilitation of the Shoulder Complex. *Applied Sciences* **2022**, *12*.
2. Kinetec, Alex exoskeleton. http://www.wearable-robotics.com/kinetek/products/alex/.
3. Hocoma Website. https://www.hocoma.com/solutions/armeo-power/.
4. Zeiaee, A.; Soltani-Zarrin, R.; Langari, R.; Tafreshi, R. Design and kinematic analysis of a novel upper limb exoskeleton for rehabilitation of stroke patients **2017**. pp. 759–764. doi:10.1109/ICORR.2017.8009339.
5. Zimmermann, Y.; Sommerhalder, M.; Wolf, P.; Riener, R.; Hutter, M. ANYexo 2.0: A Fully-Actuated Upper-Limb Exoskeleton for Manipulation and Joint-Oriented Training in all Stages of Rehabilitation. *IEEE Transactions on Robotics* **2023**. doi:10.3929/ethz-b-000584411.
6. Kim, B.; Deshpande, A.D. An upper-body rehabilitation exoskeleton Harmony with an anatomical shoulder mechanism: Design, modeling, control, and performance evaluation. *The International Journal of Robotics Research* **2017**, *36*, 414 – 435.
7. Pignolo, L.; Dolce, G.; et Al., G.B. Upper limb rehabilitation after stroke: ARAMIS a "robo-mechatronic" year=2012, volume=, number=, pages=1410-1414, doi=10.1109/BioRob.2012.6290868. 2012 4th IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob).
8. Otten, A.; Voort, C.; Stienen, A.; Aarts, R.; van Asseldonk, E.; van der Kooij, H. LIMPACT:A Hydraulically Powered Self-Aligning Upper Limb Exoskeleton. *IEEE/ASME transactions on mechatronics* **2015**, *20(5)*, 2285–2298.
9. Nef, T.; Mihelj, M.; Kiefer, G.; Perndl, C.; Muller, R.; Riener, R. ARMin - Exoskeleton for Arm Therapy in Stroke Patients. 2007 IEEE 10th International Conference on Rehabilitation Robotics, 2007, pp. 68–74. doi:10.1109/ICORR.2007.4428408.
10. Proietti, T.; Crocher, V.; Roby-Brami, A.; Jarrasse, N. Upper-Limb Robotic Exoskeletons for Neurorehabilitation: A Review on Control Strategies. *IEEE Rev Biomed Eng.* **2016**, *9*, 4–14.
11. dSPACE Website. https://www.dspace.com/en/pub/home/applicationfields/portfolio/bussimulation/bussimulation_usecases/rapid_control_prototyping.cfm.
12. Speedgoat RCP Website. https://www.speedgoat.com/solutions/simulink-real-time-workflow.
13. National Instrument Website. https://www.ni.com/docs/en-US/bundle/labview-control-design-and-simulation-module/page/lvsimconcepts/sim_c_config.html.
14. Opal-RT Website. https://www.opal-rt.com/rapid-control-prototyping/.
15. ETAS Website. https://www.etas.com/en/company/realtimes-2021-next-generation-large-scale-prototyping.php.
16. Quanser Website. https://www.quanser.com/products/quanser-rapid-control-prototyping-toolkit-labview/.
17. Bodo, G.; Tessari, F.; Buccelli, S.; Guglielmo, L.D.; Capitta, G.; Laffranchi, M.; Michieli, L.D. Customized Series Elastic Actuator for a Safe and Compliant Human-Robot Interaction: Design and Characterization. *IEEE Int Conf Rehabil Robot. 2023.* **2023**.
18. Speedgoat Website. https://www.speedgoat.com/products-services/real-time-target-machines/baseline-real-time-target-machine.
19. Speedgoat Communication Protocols Website. https://www.speedgoat.com/products-services/real-time-target-machines/baseline-real-time-target-machine.

 doi:10.20944/preprints202402.0744.v1

20. Reddy, A.C. DIFFERENCE BETWEEN DENAVIT - HARTENBERG (D-H) CLASSICAL AND MODIFIED CONVENTIONS FOR FORWARD KINEMATICS OF ROBOTS WITH CASE STUDY. *International Conference on Advanced Materials and manufacturing Technologies (AMMT)* **2014**, pp. 267–286.
21. ptc® Website. https://www.ptc.com/it/products/creo.
22. Hogan, N. Impedance control: An approach to manipulation. *ASME Journal of Dynamic Systems, Measurement, and Control* **1985**, *107*, 1–24.
23. Winstein, C.J.; Kay, D. Chapter 16 - Translating the Science into Practice: Shaping Rehabilitation Practice to Enhance Recovery after Brain Damage. In *Sensorimotor Rehabilitation At the Crossroads of Basic and Clinical Sciences*; Elsevier, 2015; Vol. 218, *Progress in Brain Research*, pp. 331–360.