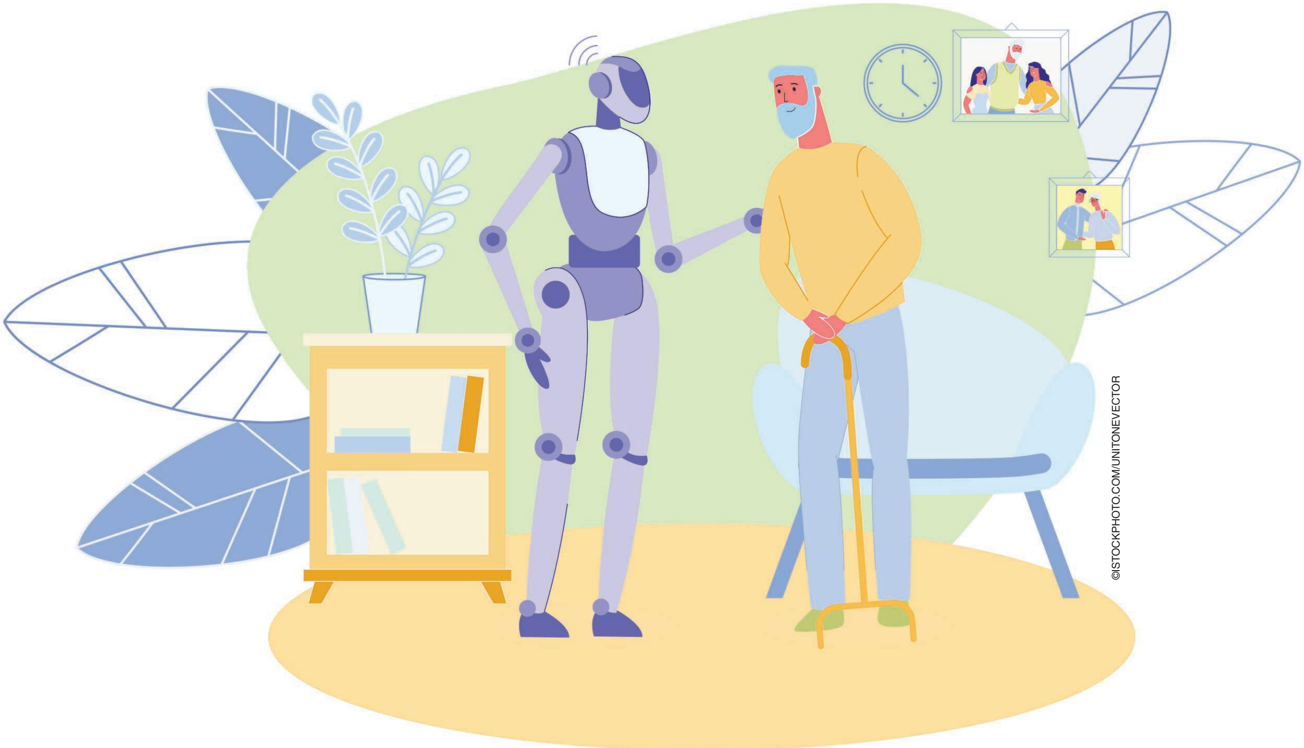
Compliant Humanoids Moving Toward Rehabilitation Applications

Transparent Integration of Real-Time Control, Whole-Body Motion Generation, and Virtual Reality



©ISTOCKPHOTO.COM/UNITONEVECTOR

By Pouya Mohammadi, Enrico Mingo Hoffman, Niels Dehio, Milad S. Malekzadeh, Martin Giese, Nikos G. Tsagarakis, and Jochen J. Steil

Humanoids fascinate us through their anthropomorphic appearance, high dexterity, and potential to work in human-tailored environments and interact with humans in novel applications. In our research, we promote two real-world applications in physiotherapeutic juggling and assisted walking using two compliant humanoids (COMANs), COMAN and COMAN+. We focus on rehabilitation, which, as a result of changing demographics, is becoming an increasingly crucial application field. However, as with most humanoid experiments, the realization of these scenarios is challenging because the hardware is brittle, the software is complex, and control remains highly demanding. In this article, we describe an integrative and transparent control architecture that alleviates this complexity by strictly adhering in design and implementation to a component-based approach. It promotes flexibility and reusability and

Digital Object Identifier 10.1109/MRA.2019.2940970
Date of current version: 21 October 2019

allows transparent switching among different humanoid robots, between simulation and the real world, and among control paradigms. It also orchestrates the integration of real-time (RT) and non-RT (NRT) components, including a virtual reality (VR) framework, toward rich user interaction.

The Humanoid Robotics Decathlon

Humanoid robotics is a fascinating field that has captured the imaginations of researchers, artists, and the general public. Endowing humanoids with the ability to execute tasks in the real world, however, still poses enormous challenges. Indeed, it represents the decathlon of robotics in that it requires mastery of diverse fields and the integration of expertise from different disciplines to achieve a generic humanoid robot assistant. Likewise, simpler platforms could typically perform specific tasks more reliably, just as a specialized athlete performs better in a single discipline than a decathlete.

In practice, most humanoid robots are still prototypic research platforms that are rather fragile, expensive, and far from productive for everyday use. Actuator technologies are still developing; control paradigms, including position and torque control, differ; dynamic walking must be further addressed; and the robots' high dexterity is based on a high degree of redundancy, which, in turn, requires that task hierarchies be defined to make use of and resolve this [1]. This comes in addition to challenges in integration with perception and state estimation, user interaction, planning, and so on.

Consequently, most control systems are highly tailored to particular hardware and specific tasks. Algorithms can hardly ever be benchmarked because of the difficulty of switching between hardware platforms. System integration is often particularly difficult as it is trying to hit a moving target, namely, the integration of a number of continuously and rapidly changing technologies, including actuation and sensor hardware, RT bus systems and protocols, component models, robotics control libraries, advanced control algorithms, and diverse operating systems, to name only a few. This calls for smart engineering and intermediate steps toward more robustness, repeatability, and flexibility.

In our research, we approach two real-world applications, the problems of physiotherapeutic juggling and assisted walking, using the compliant humanoid robots COMAN and COMAN+ (see Figure 1 and Table 1). This is strongly motivated by the demographic changes that many countries are facing. By 2020, for instance, a quarter of the European Union's population will be older than 60 years [2], leading to increased demand for physical training, rehabilitation, and assistance.

The contribution of this article is to promote a systematic, model-based approach to control architecture design that mediates some of the described complexities of such applications. Our design follows the principles of modularity and

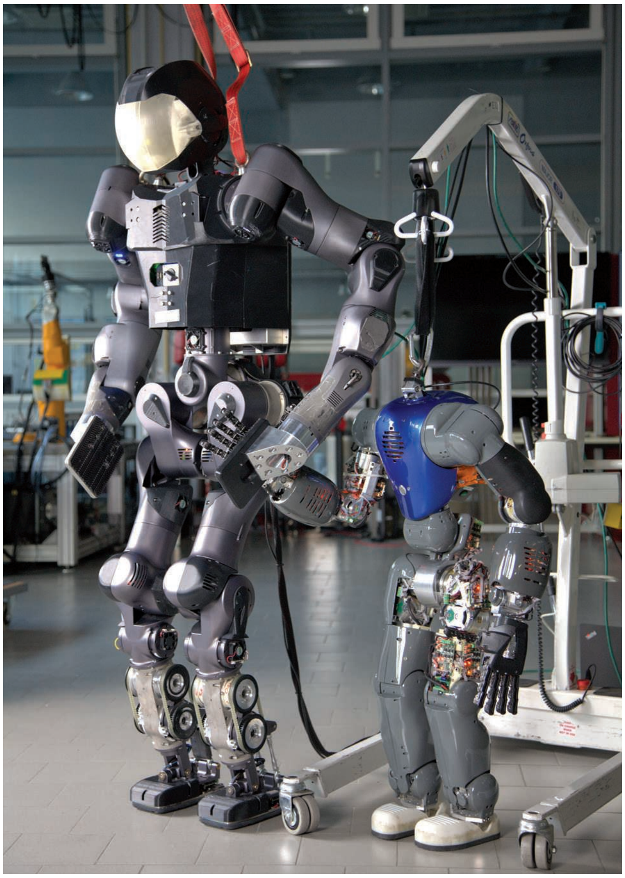


Figure 1. The COMAN (right) and its scaled-up version COMAN+ (left) humanoid robots. Both robots were used to evaluate the model-based approach to control architecture design of this article.

Table 1. The specifications for the COMAN and COMAN+ platforms.

Parameters	COMAN	COMAN+
Degrees of freedom		
Legs	6	6
Arms	7	7
Torso	3	2
Neck	—	—
Modes	Position, impedance, voltage, torque	Position, impedance, torque
Mass (kg)	35	70
Height (m)	0.95	1.7
Software	OROCOS, ROS	XBotCore, OROCOS, ROS
Sensors	IMU 4 × F/T 6 axes Link and joint side encoders Joint-torque sensors	IMU 4 × F/T 6 axes Link and joint side encoders Joint-torque sensors Lidar

F/T: force/torque; IMU: inertial measurement unit.

separation of concerns from software engineering to integrate functionality in an RT safe environment with the aim of providing a blueprint for a reusable, hardware-independent system. It features transparent switching between robot and simulation, between different hardware or control paradigms, and the assimilation of NRT components, such as VR. It thereby supports a flexible but systematic application development while accommodating diverse and changing technologies. We focus mainly on the functional architecture and control of real and simulated humanoid robots COMAN and COMAN+ in the context of the following real-world scenarios.

Application of VR in Physiotherapy

Although some humanoid robots, such as our COMAN and COMAN+, have become inherently safer through the employment of physical compliance, they are still too complex to safely interact with users outside the lab. However, the advent of inexpensive and stable VR systems provides new opportunities for intermediate steps toward user interaction. We propose invoking the actual model-based, real-world controller to drive a rendered robot in VR that interacts with the user through standard VR interfaces. This approach immediately tackles cost and robustness liabilities, while development, testing, and deployment of the control system can be enhanced through engaging in much more complex tasks. This makes the interaction inherently safer and richer, and it mediates the necessity of advanced perception, which is partially replaced through the VR tools. It is, however, necessary to ensure realistic robot physics and motion dynamics to keep the user engaged and prevent jeopardizing the user experience.

This integration of VR and humanoid robotics was originally motivated by the concrete rehabilitation application of physiotherapeutic juggling [3], where the patient and therapist mutually catch and throw a lightweight ball. The exercise is an essential part of the treatment for many age-related conditions, such as rehabilitation from stroke. It is motivating and demanding [4], [5] and improves coordination and balance [6] since it requires synchronous arm movement and posture control. Furthermore, it can enhance the arm motion and trunk–arm coordination of patients with Parkinson's disease [7], [8].

Joint-Assisted Walking

In our second scenario, we approached the problem of joint-assisted walking. Here, the humanoid robot assumes the role of the follower and attempts to react to perceived human intentions. Such advanced interaction will be necessary both for assistance in manipulation of large objects and in rehabilitation through assisted walking. It requires an intention detection algorithm, which is realized through assessing manipulability in our approach.

System Requirements

To mediate the technical difficulty of robotic experiments in general and in humanoid robotics in particular, it is crucial to devise system architectures that alleviate complexity, facilitate development, and grant scalability. This section discusses the respective requirements.

Safety

The system must be safe for the users at all times and under all conditions; however, in an open-ended user interaction, unpredicted actions can occur. To enable a physiotherapy application, we use VR to mediate critical cases as there is no direct physical interaction with the robot and only highly controlled physical interaction through a haptics interface. A sufficient safety margin can be enforced for the robot by mechanical restraints, such that it cannot endanger a patient even in the case of a technical failure.

RT

RT coordination is paramount to realizing the typical control schemes for dynamic stability that rely on precisely timed sensing of the robot state (such as encoders, the inertial measurement unit, force/torque sensors, and so on) and communication between components. This holds even for simulated robots [9]. In VR-based physiotherapy and similar user interactions, there is the further important aspect that unhandled delay can quickly jeopardize the user experience, as humans perceive visual clues as fast as 13 ms [10].

On the other hand, there are typically many components that do not rely on (strict) RT, such as motion planning, long-term memory, or other capabilities integrated with the Robotic Operating System (ROS). Thus, another requirement of flexible RT control of humanoids is to isolate the NRT components so that their lack of reliable response time does not impede the core execution of RT components.

Secondary Requirements to Support Flexibility and Reusability

Given the engineering efforts dedicated to the design and implementation of humanoid robotic control systems, it is understandable that flexibility and reusability are desired.

System transparency: The ability to switch between different robots with a minimum of programming effort is generally desired. Although there are platform-specific parameters to be tuned (for example, control gains), proper model-based abstractions can relieve the burden of switching to a considerable degree. This includes reducing transparent switching between the real and simulated robots [9] to simple plug-and-play components because simulation is typically an indispensable part of the development and testing process.

Reusability: The reuse of functionality across applications is a persistent goal of robotic software development, and it supports the repeatability of experiments. Specifically, the ability to deploy third-party algorithms can drastically reduce the cost and complexity of the development.

Adaptability to emerging technology: Another often overlooked aspect of reusability is adaptability to new and emerging technologies. Improved algorithms, faster and more reliable solvers, and new middleware frequently become available; hence, a modern architecture must be able to integrate them with reasonable effort. The initial investment of time and development in architecture design will eventually pay off when it can be migrated to keep up with changing technology demands.

Architecture

To address the requirements, we have realized the software architecture CoSiMA (Compliant Simulation and Modeling Architecture [9]), which is developed through the European joint project CogIMon [18]. We summarize it only briefly to provide the basis for discussing the specific functional properties for humanoid robotic applications.

We use Open Robot Control Software (OROCOS) [11] as the underlying RT component framework because components intrinsically support reusability in providing self-sustained and deployable units of computation (as opposed to classes). Specifically, the OROCOS Component Library provides the reusable functionality, whereas the deployment is handled by the OROCOS Deployment Component and the Deployer Environment.

The RT requirements are handled by the Real-Time Toolkit (RTT), which is an integral part of OROCOS. The RTT exposes ports, handles their communication, and provides tools to create new typekits to transfer dedicated data structures over these ports. The isolation of the NRT components, their synchronization with the RT components, tracking of delay or latency, and management of unexpected delays are resolved by the robotics service bus (RSB) [12] middleware or RTT–ROS integration.

To address the secondary requirements, a larger number of specific functional components are provided at different granularity levels and operational objectives.

Robot models and interfaces: The exposure of identical interfaces and ports across real hardware and simulation as well as different robots is achieved through a rigorous model-based approach that resorts to common abstractions based on robot unified robot description format (URDF) and semantic robot description format (SRDF) descriptions (Figure 2). These encapsulate all necessary information including kinematic structure, available sensors, actuator proportional-integral-derivative gains, and joint limits to configure the interfaces. By using SRDF, developers can also operate the robot based on conceptual entities, such as *right arm*, rather than dealing only with an *n*-vector of whole-body joint configuration. This design paradigm further alleviates the reusability of the components.

Practically, the task to read and write from/to the robot's low-level drivers and mediate and broadcast the hardware-related information is implemented in a single component named *rtt_robot* in Figure 2. It is parametrized with respect to the URDF and SRDF files and is the only system part that needs to be implemented according to the specific robot driver protocol or simulation environment.

Switching from simulation to the real robot or between robots is thus just a matter of changing this single component and is thereby reduced to the inevitable minimum of adjusting control parameters or actuator gains.

Functional and control components: Through leveraging of the component library and deployment tools of OROCOS, many components have been implemented to facilitate the development process. This includes, but is not limited to, walking-pattern generators (WPGs), stabilizers, and trajectory generation as well as loggers and helper tools. These components are deployed in RT or NRT units depending on their internal implementation.

A centerpiece is the motion engine component, where we employ a stack-of-tasks (SoT) approach that is based on inequality hierarchical quadratic programming (iHQP) [1]

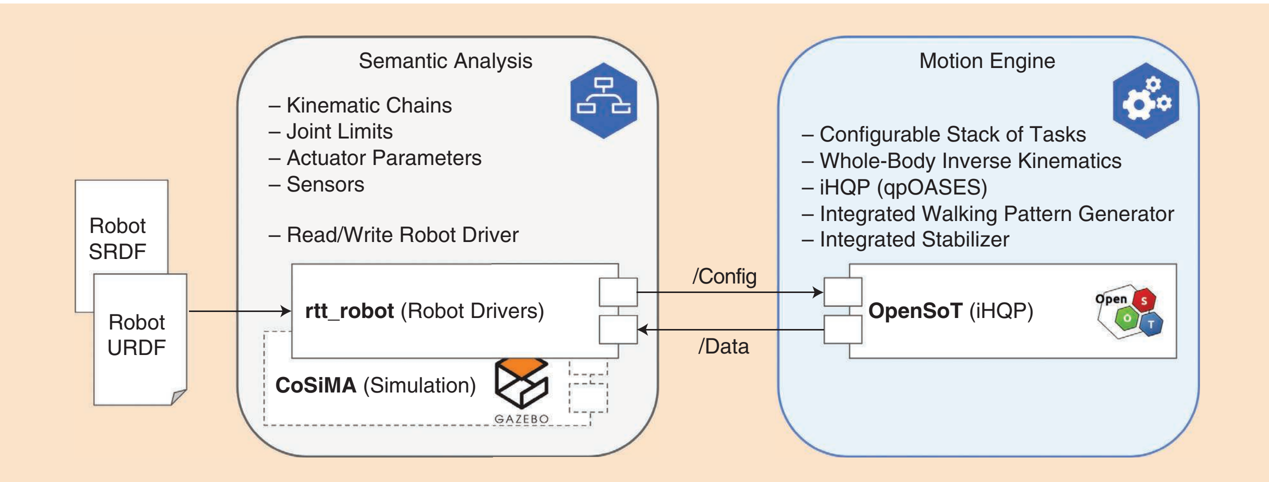


Figure 2. A semantic analysis of the robot model: kinematic and dynamic information for joints and links from URDF and semantic information, such as kinematic chains, from SRDF are parsed, grouped, and exposed in interfaces to different components including the control side. Likewise, the motion engine is configured according to the information broadcast by the component in the semantic analysis block.

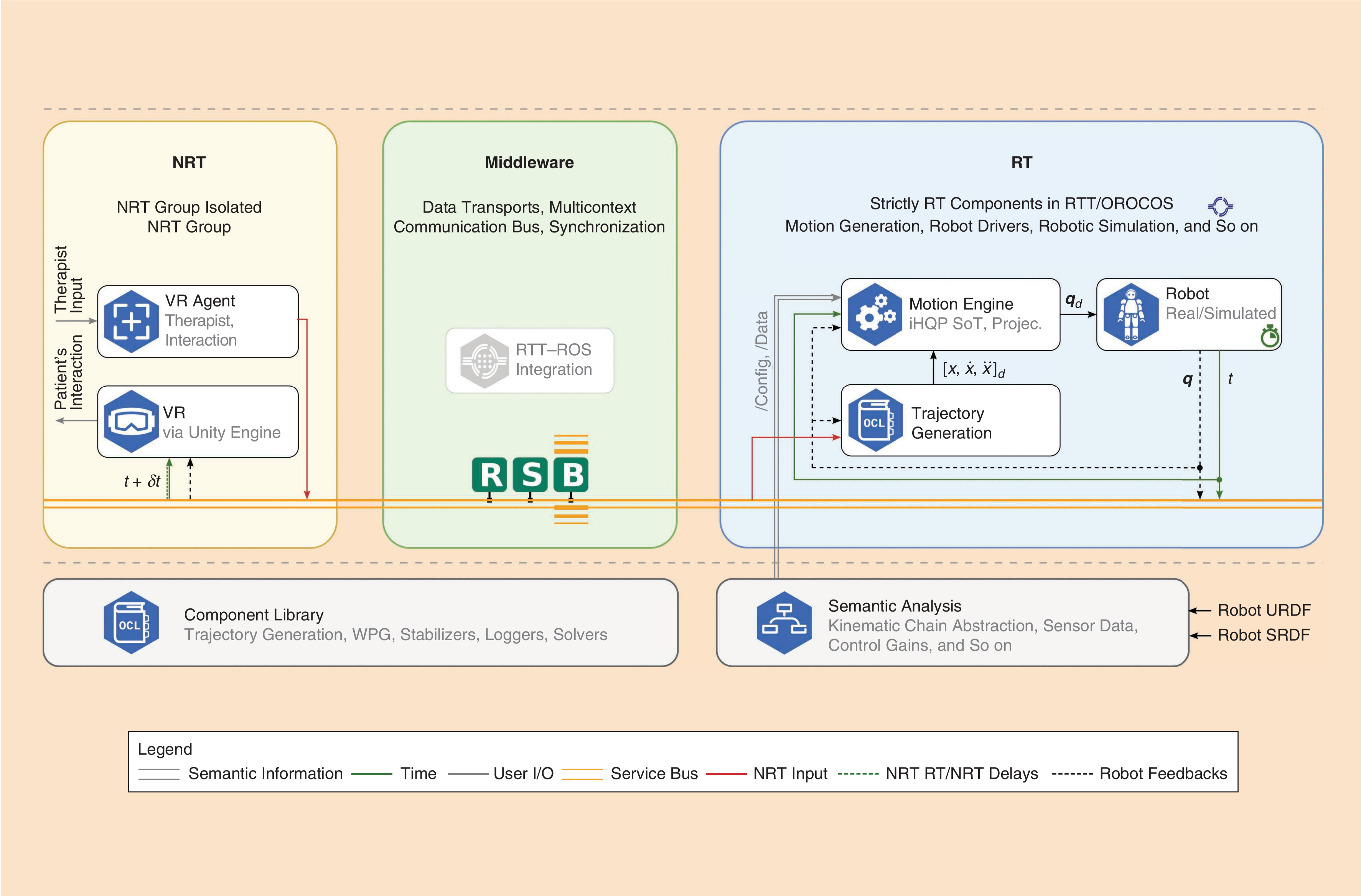


Figure 3. The functional architecture in the context of physiotherapeutic juggling. The system comprises three major units: RT and NRT units as well as a middleware that handles the communication, synchronization, and data transport between the two. In the case of juggling, the RSB plays this role; in assisted walking, RTT–ROS integration was used. Building blocks of RT and NRT units are components that constitute a component library and are deployed via OROCOS. Semantic analysis (Figure 2) provides input to different components, including the motion engine. I/O: input–output; Projec.: projection.

and formulated as a velocity-control scheme together with a joint level-impedance controller (see Figure 2). It utilizes the OpenSoT [13] software tool, which simplifies setting up and solving the respective optimization problems while adhering to the kinematic chain abstractions. OpenSoT enables the developer to combine tasks and constraints as atomic entities without explicit definition of Jacobians or interaction with the kinematic or dynamic solvers.

For the sake of brevity, we focus only on the SoT, but we can alternatively also employ dynamically consistent task hierarchy projections [14] as the motion engine. The latter resorts to computed torque control, thereby switching the control paradigm substantially within the same overall architecture.

For illustration, let us provide a bird's-eye view of the described architecture by considering a single iteration of throwing a ball in the physiotherapy scenario with reference to Figure 3. Based on specific needs of the patient, the therapist selects a throw point for the ball via the VR agent in the NRT unit. This point is communicated to the trajectory generation component in the RT unit via the robot service bus. The component computes an end-effector trajectory for the arm of the robot, with which it realizes the desired landing point of the ball and sends its results to the motion engine over OROCOS ports. After the SoT is solved, the desired joint motions are sent to the robot simulator. The robot's feedback is sent directly to the other components in the RT unit; in parallel, they are sent to the NRT unit through the RSB, where the robot is rendered for the patient in the VR component.

Communication between the RT and the NRT units is handled strictly by RSB middleware. Furthermore, the robot or simulator clock is considered as the reference clock, although intrinsically it is provided by the OROCOS framework. The middleware provides mechanisms to measure or handle delays (for example, due to network latency).

SoT for Hierarchical Quadratic Programming

We follow a standard hierarchical SoT approach [1], which is typical in humanoid robot control due to the high number of degrees of freedom. Lower-priority tasks are executed in the null space of higher priority tasks to ensure that important tasks are not compromised, and less important tasks are followed only to a degree, depending on the possibly varying availability of remaining null-space motion. Additionally, inequality constraints, such as joint position and velocity limits, are integrated. We discuss our hierarchy of hard and soft priorities and the definition of major equality and inequality tasks following the formalism of OpenSoT [13].

Denote the generalized coordinates of a humanoid robot by $\boldsymbol{q} \in \mathbb{R}^{6+n}$. The first six coordinates represent the underactuated virtual chain, attached from the inertial frame to the floating base, while the remaining n are associated with the actuated joints. Tracking the 6D Cartesian position and orientation of a frame F attached to an arbitrary operational point of the robot constitutes a task,

$$\mathcal{T} = \langle \boldsymbol{J}, \dot{\boldsymbol{x}}^{*} + \lambda \boldsymbol{e} \rangle, \quad (1)$$

where $\boldsymbol{e} \in \mathbb{R}^6$ is the Cartesian pose error between F and desired F^*, $\dot{\boldsymbol{x}}^* \in \mathbb{R}^6$ is the desired velocity of F, $\boldsymbol{J} \in \mathbb{R}^{6 \times 6+n}$ is the task Jacobian, and λ is a positive scalar gain. The task can be associated with a constraint $C = \langle \boldsymbol{A}, \boldsymbol{b} \rangle$, where matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$ define the solution boundaries.

At priority level i, the quadratic form of this task, along with its constraint, is expressed as

$$\begin{aligned} \mathcal{T}_i &:= \dot{\boldsymbol{q}}_i = \underset{\dot{\boldsymbol{q}} \in \mathcal{S}_i}{\operatorname{argmin}} \frac{1}{2} \| \boldsymbol{J}_i \dot{\boldsymbol{q}} - \dot{\boldsymbol{x}}_i^* - \lambda \boldsymbol{e} \|^2 + \epsilon \| \dot{\boldsymbol{q}} \|^2 \\ C_i &:= \boldsymbol{A}_i \dot{\boldsymbol{q}} \leq \boldsymbol{b}_i, \end{aligned} \quad (2)$$

where $\dot{\boldsymbol{q}}$ are the joint velocities and ϵ is a regularization term. $\mathcal{S}_i$ is the set of all possible solutions at this priority level and

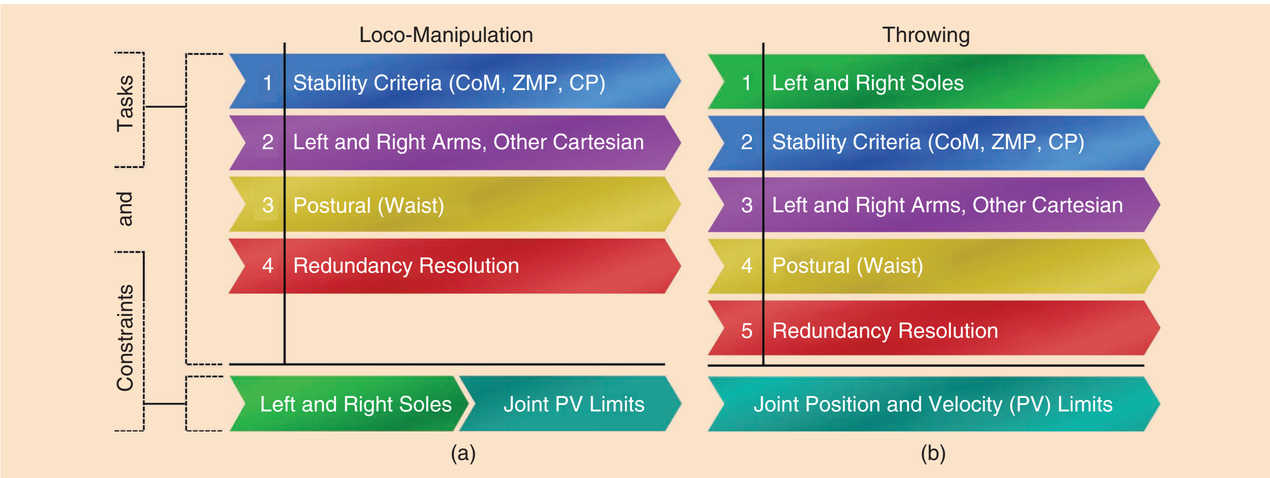


Figure 4. The priorities for (a) assisted walking and (b) physiotherapeutic juggling. The sole contacts have the highest priority in throwing, whereas, in loco-manipulation they are inserted as constraints to achieve better computational performance. Next is the center-of-mass (CoM) task, followed by manipulation, postural, and redundancy tasks. The constraints of these task are joint position and velocity limits. ZMP: zero moment point; CP: capture point.

lies in the null space of the task $\mathcal{T}_{i-1}$ at priority level $i-1$. Thus, the volume of $\mathcal{S}_i$ shrinks as i increases and the priority decreases. Consequently, for two tasks, the solution of a task $\mathcal{T}_j$ must reside in the null space of $\mathcal{T}_i$ if $j > i$, that is, they have "hard" prioritization.

Alternatively, it is possible that two tasks share the same priority level, hence forming a "soft" priority. This is done by augmenting the Jacobian and Cartesian velocities of these tasks. Soft prioritization between the tasks can be achieved using a weighting. An example of soft priorities is the contact tasks for the left and right soles of a humanoid, where it is crucial that both positions and orientations are tracked accurately. In the SoT of our scenarios, this task has the highest priority [Figure 4(a)], or it is injected as the constraint of the highest-priority task ([Figure 4(b)].

In the null space of the contact tasks, the motion of the center of mass (CoM), which is responsible for the balance, is tracked. In the throwing scenario, where the robot is not stepping, a stabilizer computes suitable CoM trajectories with the goal of bringing the zero moment point (ZMP) to the center point of the convex hull of the feet contact polygon. When stepping in assisted walking, a pattern generator creates dynamically consistent CoM motions according to the linear inverted pendulum dynamics and the reference CoM velocity governed by

$$\dot{\boldsymbol{x}}_{\text{CoM}} = \begin{cases} \dfrac{\boldsymbol{u}_m^{\text{L}}}{\sigma_m^{\text{L}}} + \dfrac{\boldsymbol{u}_m^{\text{R}}}{\sigma_m^{\text{R}}}, & \text{if } \ (\nu^{\text{R}} \le \nu_d) \vee (\nu^{\text{L}} \le \nu_d), \\ 0, & \text{if } \ (\nu^{\text{R}} > \nu_d) \wedge (\nu^{\text{L}} > \nu_d). \end{cases} \tag{3}$$

In (3), we monitor the volume of the manipulability ellipsoid of the left ν^{L} and the right ν^{R} arms of the COMAN+. When either hits a threshold ν_d, the desired $\dot{\boldsymbol{x}}_{\text{CoM}}$ is computed based on left singular vectors $\boldsymbol{u}_m$ associated with the smallest singular values σ_m, obtained from singular value decomposition of the arms' Jacobians (Figure 5).

On the next level, it proves useful to assign a task to the waist that maintains a suitable Cartesian orientation and prevents rotations along the roll and pitch with respect to the inertial frame but allows yaw (along craniocaudal or the z-axis) rotations. In throwing, this potentially helps the robot toss the ball farther, whereas, in walking, it can align the upper and lower bodies.

At the lowest priority, any remaining redundancy is resolved through a task $\mathcal{T}_N$ that expresses preference for some default motion $\boldsymbol{q}_N^*$ and $\dot{\boldsymbol{q}}_N^*$—for example toward a home position—with task Jacobian as identity matrix $\boldsymbol{I}_n \in \mathbb{R}^{n \times n}$:

$$\mathcal{T}_N = \langle \boldsymbol{I}_n, \dot{\boldsymbol{q}}_N^* + \lambda(\boldsymbol{q}_N^* - \boldsymbol{q}) \rangle. \tag{4}$$

Joint position and velocity limits are applied at the top priority levels as inequality constraints, whereas, in (2), we have $\boldsymbol{A} = \boldsymbol{I}$ and $\boldsymbol{b} = \{\dot{\boldsymbol{q}}^+, \dot{\boldsymbol{q}}^-\}$ for joint velocity limits. Joint position limits are similarly handled through a numerical integration.

Figure 4 shows the overall SoT. Its structure follows a straightforward logic: stability and balance criteria are of the highest importance, the manipulation task comes at the next

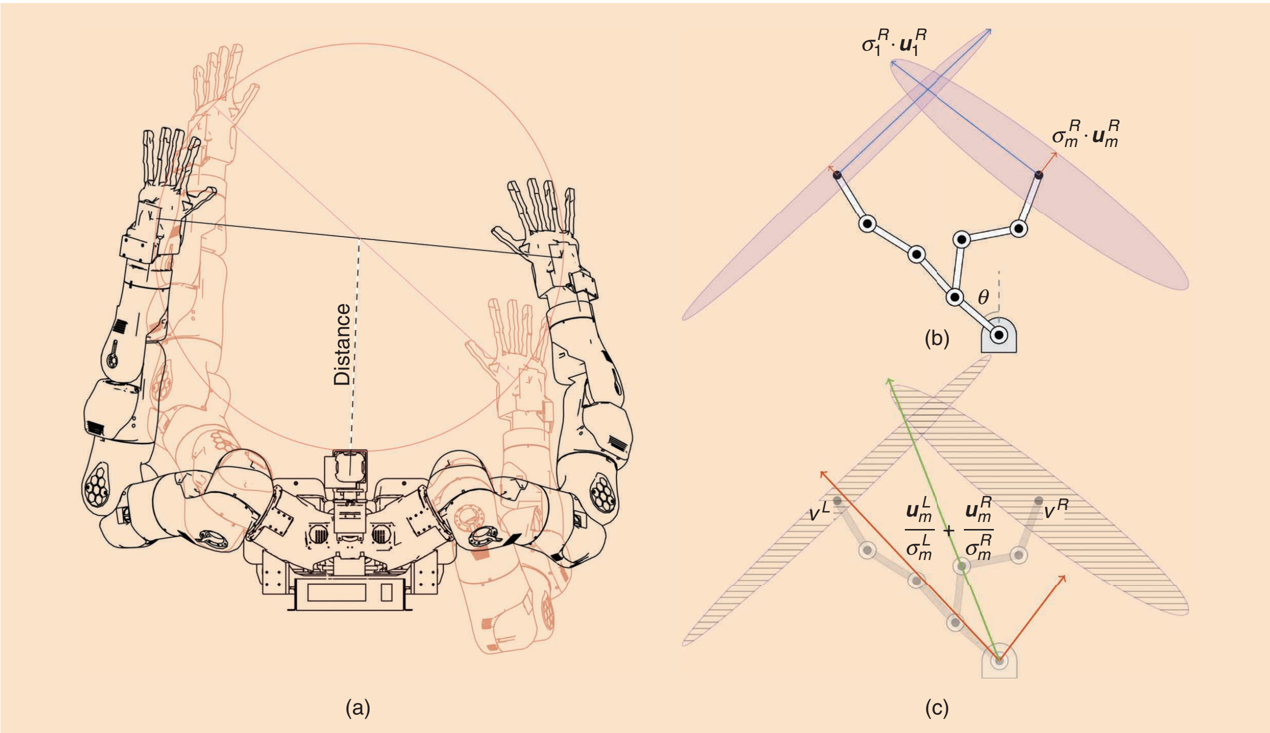


Figure 5. (a) The black and red configurations have identical distances to the torso, but red requires a left–forward step. This demonstrates that distance is not sufficient to determine stepping direction. (b) The manipulability ellipsoids of the simplified model with semi-axes. The red axes show the singular vectors for the smallest singular values. (c) The axes are scaled by $1/\sigma_m$. The sum indicates the walking direction (green).

priority level, and the remainder is designed to achieve secondary objectives, such as human-like motion.

Experimental Validation

Physiotherapeutic-Juggling Scenario

The data flow of a single iteration of a throw was described in the "Architecture" section. Here, we provide results of the first empirical evaluations, initially tested in the laboratory with healthy staff (who were familiar with both the robot and the VR) and later with patients. The experimental setup validated the RT and NRT integration together with the reliability of the SoT-based motion generation. Subsequently, we exposed patients to the system and evaluated their feedback regarding comfort level, user satisfaction, and practical feasibility. The patients' preliminary feedback was highly positive; they were able to catch balls and showed the desired training effects. Figure 6 shows a patient and his view into the VR. We are in the process of obtaining ethical permissions for a systematic evaluation at the time of this writing.

Technically, the robot simulator component, including Gazebo, was running at a 2-ms cycle time and achieved a 99% RT factor while using Open Dynamics Engine as the physics engine. The dynamically simulated motions were rendered in the VR engine Unity for the patient wearing goggles. Although the juggling exercises in VR turned out to be a standalone application on its own merits, for the sake of completeness, we address the possibility of performing them on the real robot. Based on our experiments, the real COMAN is

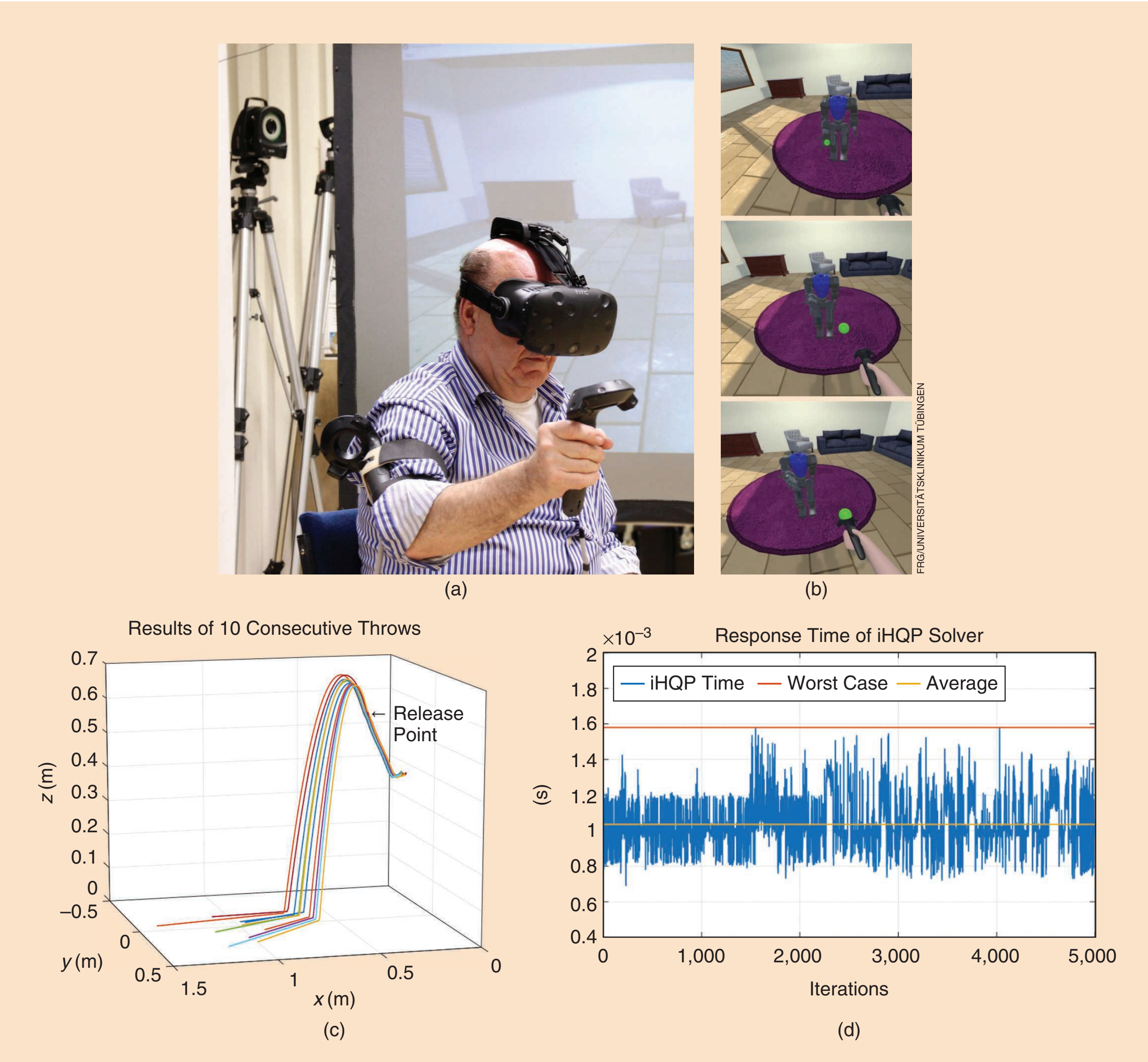


Figure 6. The experimental results from physiotherapeutic juggling in VR. (a) A volunteer patient performing the catching exercise. (b) Some snapshots of the view through the VR goggles. (c) The results of 10 consecutive throws. (d) The response time of the iHQP solver.

capable of generating the appropriate motions to make throws; however, the hand and its fingers are not yet fast enough to release the ball in a timely manner.

Assisted-Walking Scenario

Here, we present experimental validation of a close interaction between the robot follower and human leader. At each control cycle, the change of manipulability resulting from hand-in-hand interactions between the user and the robot dictates the walking direction according to (3) and Figure 5.

The joint references computed by the whole-body inverse kinematics, inside the motion engine, are sent to a decentralized joint-level impedance control. The joint stiffness in both arms is kept low to guarantee enough compliance with the human patient. As seen in Figure 4, it uses a similar iHQP-based SoT as the previous scenario with the addition of a WPG based on [15]. The overall control scheme is depicted in Figure 7.

In this scenario, a hard RT constraint must be enforced due to the nature of the experiment. The SoT was running at 100 Hz, and the WPG was running at 10 Hz, achieving RT performance under a Xenomai patched environment. Calculation of walking direction based on manipulability was performed in an NRT component developed as an ROS node. The performance of the SoT controller that concerns the tracking of the CoM and feet computed by the WPG is shown in Figure 8, while implementation and technical aspects are detailed in [16].

The setup for this second scenario highlights the transparency toward technology scale-up in two regards. First, RSB was replaced by RTT–ROS integration for NRT communications. More important, the COMAN+ hardware drivers are based on a new framework called *XBotCore* [17]. Nonetheless, exposure of identical interfaces encapsulated this technology change from the developers. Furthermore, the WPG was also tested on the COMAN robot without any programming efforts, even though the robots have different kinematic and dynamic structures. A complementary video attachment for both of the experiments is available at https://youtu.be/v2zFpngoe_Q.

Conclusions

We presented two humanoid interaction scenarios with real-world applications and implications, focused on physiotherapy and rehabilitation. An architecture that features a strict separation of concerns proved to be paramount for their swift development, which shows a very high degree of reusability. Through strictly adhering to model-based abstractions, identical interfaces for the robots and their simulated versions are exposed to the developers, allowing seamless switching between them. This was lifted to an even higher tier by providing semantic abstractions in terms of kinematic chains, encapsulating underlying structural differences.

The integration of VR engages users, strongly enhances interaction opportunities, and mediates safety concerns. Thanks to advancements in computer graphics, unbiased and physically consistent ray-tracing algorithms, and hardware that can handle them at a very low price, we demonstrate that roboticists can now test their controllers and explore user interactions with humanoid robots in advanced scenarios. In the VR-based physiotherapy scenario, we devised a juggling application that has already been tested with patients and in a real physiotherapy practice. Quantitative evaluation is on the way.

The assisted-walking experiment—although still in a preliminary state and heavily under development—provided insight about the nature of nonverbal interactions and showed how interaction forces can be interpreted for intention detection. Given that exposing elderly patients to humanoids for assisted walking is not yet reasonable due to safety considerations, for the time being, we are working to make our algorithms more

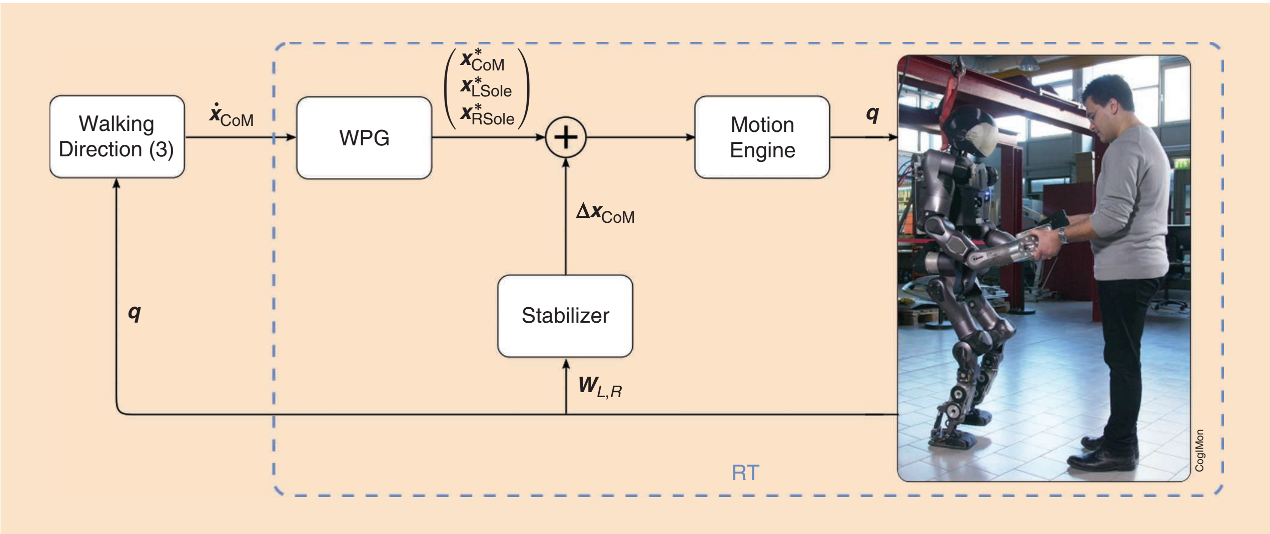


Figure 7. A schematic of the assisted-walking scenario. Feedback from the robot consists of joint positions and velocities as well as wrenches computed from ankle-mounted force/torque sensors. The former is used to computed the desired walking direction, where the outcome is sent to the WPG, and the latter is fed to the stabilizer, which corrects the WPG output.

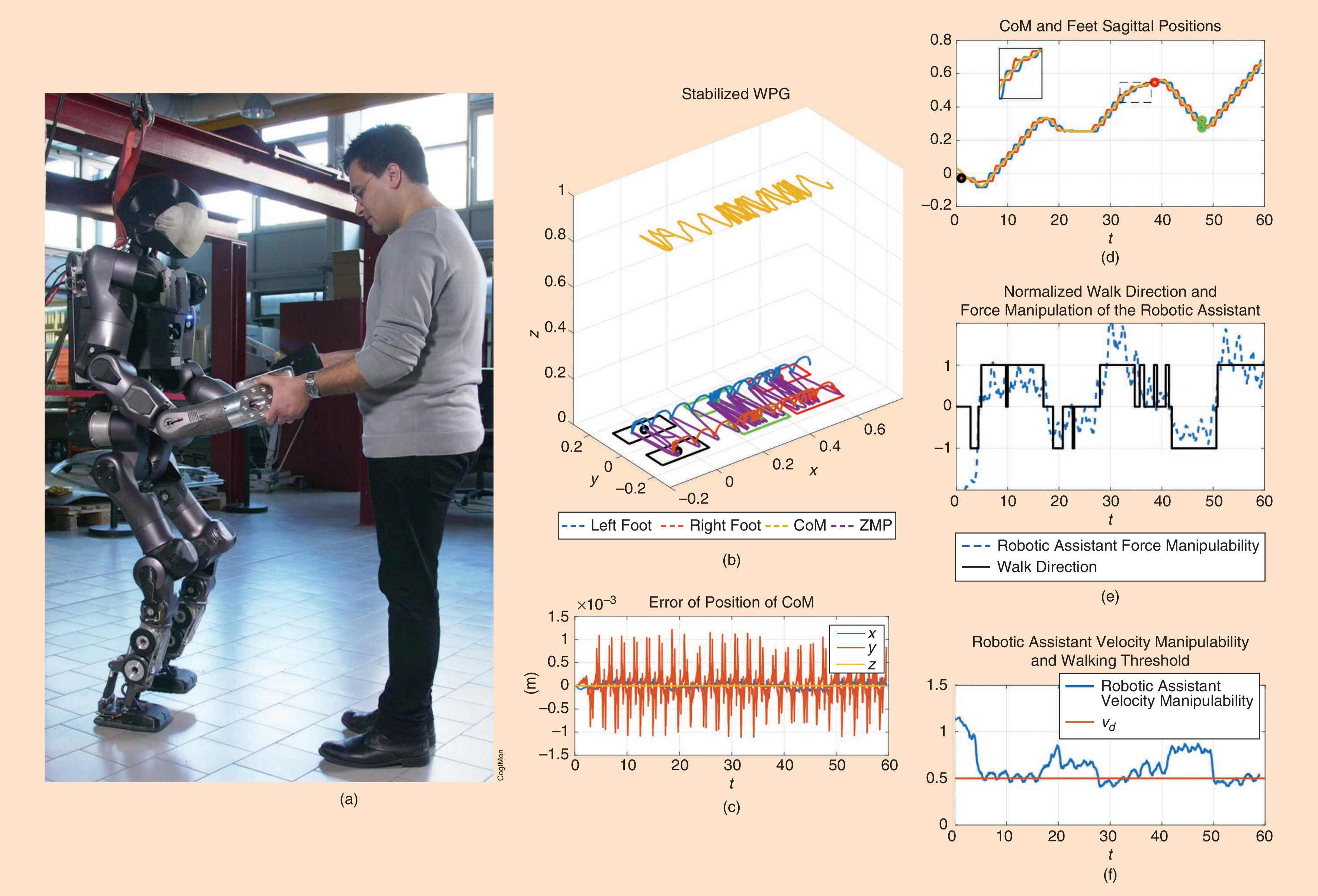


Figure 8. (a) The experimental setup for assisted walking in a human–robot loco-manipulation scenario. The user interacts with the impedance-controlled upper body, which computes and performs suitable steps to address the human partner's intentions based on its own arms' manipulability. (b) The trajectories for the ZMP, CoM, and feet of the robot. (c) The tracking error for the CoM. The (d) feet placement and CoM trajectories on the sagittal plane, (e) computed walking direction, and (f) changes of manipulability of the right arm during a 60-s experiment.

reliable. A well-designed architecture strongly enhances safety, as systematic testing is greatly facilitated.

Once humanoid robots become more reliable and affordable, they have the potential to actively support individuals through training, rehabilitation, and assistance with special needs. In physiotherapeutics, for instance, therapists will be able to orchestrate treatment plans for multiple patients, while robots will do the repetitive "muscle work." This is an exciting future that we look forward to.

Acknowledgments

The research was funded under H2020 GA 644727–CogIMon. We thank D. Brötz and J. Kodl for supervising the patient work and Y. You, C. Zhou, J.A. Castano Pena, and L. Muratore for supporting the COMAN+ experiments.

References

[1] A. Escande, N. Mansard, and P.-B. Wieber, "Hierarchical quadratic programming: Fast online humanoid-robot motion generation," *Int. J. Robot. Res.*, vol. 33, no. 7, pp. 1006–1028, May 2014. doi: 10.1177/0278364914521306.

[2] A. Chłoń-Domińczak, I. Kotowska, J. Kurkiewicz, M. Stonawski, and A. Abramowska-Kmon, *Population Ageing in Europe: Facts, Implications and Policies,* Brussels, Belgium: European Union. doi: 10.2777/60452. 2014.

[3] P. Mohammadi et al., "Real-time control of whole-body robot motion and trajectory generation for physiotherapeutic juggling in VR," in *2018 IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, Oct. 2018, pp. 270–277. doi: 10.1109/IROS.2018.8593632.

[4] B. Draganski, C. Gaser, V. Busch, G. Schuierer, U. Bogdahn, and A. May, "Changes in grey matter induced by training," *Nature*, vol. 427, no. 6972, pp. 311–312, 2004. doi: 10.1038/427311a.

[5] C. Sampaio-Baptista, N. Filippini, C. J. Stagg, J. Near, J. Scholz, and H. Johansen-Berg, "Changes in functional connectivity and GABA levels with long-term motor learning," *NeuroImage*, vol. 106, pp. 15–20, Feb. 2015. doi: 10.1016/j.neuroimage.2014.11.032.

[6] S. T. Rodrigues et al., "Postural control during cascade ball juggling," *Percept. Mot. Skills*, vol. 123, no. 1, pp. 279–294, Aug. 2016. doi: 10.1177/0031512516660718.

[7] M. J. Majsak, T. Kaminski, A. M. Gentile, and A. M. Gordon, "Effects of a moving target versus a temporal constraint on reach and grasp in patients with Parkinson's disease," *Exp. Neurol.*, vol. 210, no. 2, pp. 479–488, Apr. 2008. doi: 10.1016/j.expneurol.2007.11.023.

[8] H.-I. Ma, W.-J. Hwang, C.-Y. Wang, J.-J. Fang, I.-F. Leong, and T.-Y. Wang, "Trunk–arm coordination in reaching for moving targets in people with Parkinson's disease: Comparison between virtual and physical reality," *Hum. Mov. Sci.*, vol. 31, no. 5, pp. 1340–1352, 2012. doi: 10.1016/j.humov.2011.11.004.

[9] D. L. Wigand, P. Mohammadi, E. M. Hoffman, N. G. Tsagarakis, J. J. Steil, and S. Wrede, "An open-source architecture for simulation, execution and analysis of real-time robotics systems," in *2018 IEEE Int. Conf. Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*, May 2018, pp. 93–100. doi: 10.1109/SIMPAR.2018.8376277.

[10] C. E. Hagmann and M. C. Potter, "Ultrafast scene detection and recognition with limited visual information," *Vis. Cogn.*, vol. 24, no. 1, pp. 2–14, Jan. 2016. doi: 10.1080/13506285.2016.1170745.

[11] H. Bruyninckx, "Open robot control software: OROCOS project," in *Proc. 2001 IEEE Int. Conf. Robotics and Automation (ICRA)*, vol. 3, pp. 2523– 2528, Feb. 2001.

[12] J. Wienke and S. Wrede, "A middleware for collaborative research in experimental robotics," in *Proc. 2011 IEEE/SICE Int. Symp. System Integration (SII)*, Dec. 2011, pp. 1183–1190. doi: 10.1109/SII.2011.6147617.

[13] A. Rocchi, E. M. Hoffman, D. G. Caldwell, and N. G. Tsagarakis, "Opensot: A whole-body control library for the compliant humanoid robot COMAN," in *Proc. 2015 IEEE Int. Conf. Robotics and Automation (ICRA)*, May 2015, pp. 6248–6253. doi: 10.1109/ICRA.2015.7140076.

[14] N. Dehio, D. Kubus, and J. J. Steil, "Continuously shaping projections and operational space tasks," in *Proc. 2018 IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, Oct. 2018, pp. 5995–6002.

[15] A. Herdt, H. Diedam, P.-B. Wieber, D. Dimitrov, K. Mombaur, and M. Diehl, "Online walking motion generation with automatic footstep placement," *Adv. Robot.*, vol. 24, no. 5–6, pp. 719–737, 2010. doi: 10.1163/016918610X493552.

[16] P. Mohammadi, E. Mingo Hoffman, L. Muratore, N. G. Tsagarakis, and J. J. Steil, "Reactive walking based on upper-body manipulability: An application to intention detection and reaction," in *Proc. 2019 International Conference on Robotics and Automation (ICRA)*, May 20–24, 2019, pp. 4991–4997. doi: 10.1109/ICRA.2019.8794309.

[17] L. Muratore, A. Laurenzi, E. M. Hoffman, A. Rocchi, D. G. Caldwell, and N. G. Tsagarakis, "Xbotcore: A real-time cross-robot software platform," in *Proc. 2017 First IEEE Int. Conf. Robotic Computing (IRC)*, Apr. 2017, pp. 77–80.

[18] "Cognitive interaction in motion," Technische Universität Braunschweig. Accessed on: June 2019. [Online]. Available: https://cogimon.eu/

Pouya Mohammadi, The Institute for Robotics and Process Control, Technische Universität Braunschweig, Germany. E-mail: p.mohammadi@tu-bs.de.

Enrico Mingo Hoffman, Advanced Robotics Department, Istituto Italiano di Tecnologia, Genoa. Email: enrico.mingo@iit.it.

Niels Dehio, Intelligente Prozessautomation und Robotik, Karlsruher Institut für Technologie, Germany. Email: niels.dehio@kit.edu.

Milad S. Malekzadeh, The Institute for Robotics and Process Control, Technische Universität Braunschweig, Germany. Email: milad.malekzadeh@gmail.com.

Martin Giese, Section for Computational Sensomotorics, Department of Cognitive Neurology, Hertie Institute for Clinical Brain Research, Werner Reichardt Centre for Integrative Neuroscience, University Clinic Tübingen, Germany. Email: martin.giese@uni-tuebingen.de.

Nikos G. Tsagarakis, Advanced Robotics Department, Istituto Italiano di Tecnologia, Genoa, Italy. Email: nikos.tsagarakis@iit.it.

Jochen J. Steil, The Institute for Robotics and Process Control, Technische Universität Braunschweig, Germany. Email: j.steil@tu-bs.de.