ExoKit: A Toolkit for Rapid Prototyping of Interactions for Arm-based Exoskeletons

Marie Muehlhaus
muehlhaus@cs.uni-saarland.de
Saarland University, Saarland Informatics Campus
Saarbrücken, Germany

Alexander Liggesmeyer
liggesmeyer@cs.uni-saarland.de
Saarland University, Saarland Informatics Campus
Saarbrücken, Germany

Jürgen Steimle
steimle@cs.uni-saarland.de
Saarland University, Saarland Informatics Campus
Saarbrücken, Germany

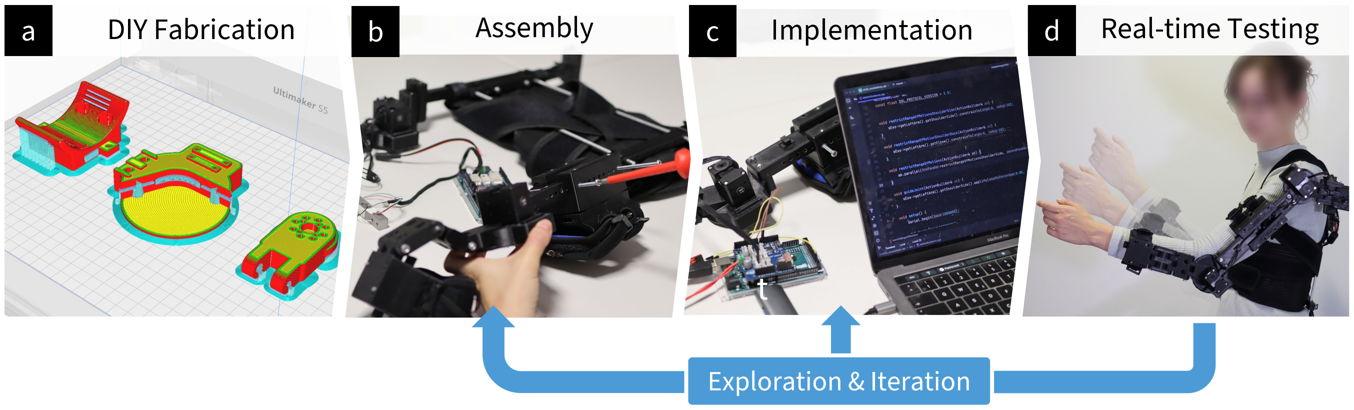

Figure 1: ExoKit enables rapid prototyping of low-fidelity exoskeletons using 3D-printed and off-the-shelf components (a). Its modular design allows easy customization of functionality and size (b). To ease customization of the exoskeleton's interactive behavior, functional abstractions of basic functionalities and higher-level augmentation strategies are readily accessible through programming interfaces (c). The designed interaction can be tested in real-time (d).

Abstract

Exoskeletons open up a unique interaction space that seamlessly integrates users' body movements with robotic actuation. Despite its potential, human-exoskeleton interaction remains an underexplored area in HCI, largely due to the lack of accessible prototyping tools that enable designers to easily develop exoskeleton designs and customized interactive behaviors. We present ExoKit, a do-it-yourself toolkit for rapid prototyping of low-fidelity, functional exoskeletons targeted at novice roboticists. ExoKit includes modular hardware components for sensing and actuating shoulder and elbow joints, which are easy to fabricate and (re)configure for customized functionality and wearability. To simplify the programming of interactive behaviors, we propose functional abstractions that encapsulate high-level human-exoskeleton interactions. These can be readily accessed either through ExoKit's command-line or graphical user interface, a Processing library, or microcontroller firmware, each targeted at different experience levels. Findings from implemented application cases and two usage studies demonstrate the versatility and accessibility of ExoKit for early-stage interaction design.

CCS Concepts

• **Human-centered computing → Interactive systems and tools**.

Keywords

Exoskeleton; interactions; toolkit; rapid prototyping; fabrication; DIY; augmented human.

ACM Reference Format:
Marie Muehlhaus, Alexander Liggesmeyer, and Jürgen Steimle. 2025. ExoKit: A Toolkit for Rapid Prototyping of Interactions for Arm-based Exoskeletons . In *CHI Conference on Human Factors in Computing Systems (CHI '25), April 26-May 1, 2025, Yokohama, Japan.* ACM, New York, NY, USA, 18 pages. https://doi.org/10.1145/3706598.3713815

1 Introduction

Exoskeletons create unique opportunities for novel interactions which assist, modify or constrain physical body movement. For instance, recent work in the HCI community has leveraged hand-based or elbow-based exoskeletons for diverse applications including motion capture [22], haptic feedback in VR [22, 64], or as a means for skill transfer between people [45].

Despite the great potential of exoskeletons, and despite decades of research in other communities, human-exoskeleton interaction is still a largely underinvestigated field in HCI. A key challenge contributing to this gap is the lack of affordable exoskeleton technologies and the lack of easily customizable and easily programmable commercial exoskeletons. The degree of expertise currently required for engineering an exoskeleton and for controlling its motion presents high barriers of entry for novices. This is further aggravated by a certain disciplinary gap in goals between HCI and robotics: while exoskeletons from robotics tend to address specific applications with sophisticated high-end technology [10, 25, 31], interaction designers need support for creatively exploring design opportunities in the early design phases. This requires low-fidelity, easily customizable and easy-to-iterate-on prototypes [52].

We address this gap with ExoKit, an **open-source toolkit for rapid prototyping of low-fidelity yet functional exoskeleton prototypes**. ExoKit is targeted at researchers, hobbyists, and makers with basic electronics and programming skills (e.g., gained through Arduino-based projects). Expertise with robotics, including their design and control, is not required[1]. ExoKit aims to support them in creatively designing and exploring novel human-exoskeleton interactions in the early design phases. ExoKit is informed by a set of considerations that we derived from the literature on HCI toolkit research, wearables and exoskeletons. The key benefits of ExoKit are: (a) *ease of development for novice roboticists*, achieved with a DIY approach that uses 3D printed components, off-the-shelf servo motors, and a dedicated Arduino firmware library, and (b) *customizability in design and behavior for rapid prototyping*, achieved through modularization of the hardware and software required for prototyping human-exoskeleton interactions. The resulting building blocks can be easily exchanged or combined into new functionalities, promoting the exploratory and iterative character of interaction design for a wide range of applications.

ExoKit's **hardware components** realize a low-fidelity exoskeleton prototype that can actuate one or two arms, with up to two active degrees-of-freedom (DoF) at the shoulder and one active DoF at the elbow. Arms hold promise for novel interactions in versatile application areas, such as physical motion guidance or strength augmentation; however, they present challenges in the mechanical design [65]. To tailor the prototype to the needs of diverse applications, the toolkit offers modular hardware components, which allow for hands-on reconfigurability of the exoskeleton's functionality, enable adjusting the size to accommodate various body sizes, and deploy important safety mechanisms.

ExoKit's **software library** provides functional abstractions which free the novice roboticist from detailed low-level motion control. We first identified frequently used augmentation strategies from state-of-the-art exoskeleton literature [14, 53] and organized them in a two-dimensional conceptual space. For each identified augmentation strategy, ExoKit provides pre-implemented functions that the developer can use off-the-shelf for rapid experimentation and ideation. Amongst others, these comprise functions that interactively amplify or resist user motion, modulate the style of a motion, transfer motion from one exoskeleton to another, or guide the user's motion through real-time haptic feedback. For the ease of development, ExoKit provides simplified access to these functions through a command-line interface, a GUI, a Processing library, or directly through the Arduino firmware. The firmware library enables the novice roboticist to compose the offered functional abstractions into more complex and meaningful interactions. We make the hardware design and software libraries openly available[2].

To confirm the versatility of the toolkit for rapid prototyping, we demonstrate application cases that have been successfully realized with the framework, discuss their iterative design process and lessons learned. Second, we present results from two usage studies. In our first study, we collect feedback of ExoKit in use, and discuss user's opinions on the provided functional abstractions, programmability, and wearability. With our second user study, we provide insights into how users approached ExoKit to create their own applications, their workflows and encountered challenges. We conclude by discussing implications for the design of human-exoskeleton interactions, limitations, and future directions.

In summary, the main contribution of this paper is ExoKit, a toolkit for rapid prototyping of human-exoskeleton interaction, targeted at novice roboticists in the early phases of interaction design. We

- conceptually identify functional abstractions that encapsulate relevant augmentation strategies, implemented in an Arduino firmware library. A designer can readily access these through a command-line interface, GUI, Processing library, or further customize the implemented strategies directly through the firmware.
- modularize the exoskeleton into hardware components that can be easily exchanged or combined into new functionalities, adjusted in size, and used to implement physical safety mechanisms. The resulting prototype can support the user with up to 3 DoF per arm.
- demonstrate ExoKit's versatility and utility through application examples and two usage studies.

We hope that this work will inspire and empower HCI researchers, designers, and makers alike to start engaging with the exciting area of human-exoskeleton interaction and exploring its potential for innovative applications.

2 Related Work

This work is informed by prior work on exoskeletons, robot programming, and toolkit research in HCI.

2.1 Exoskeletons

Robots worn on the human body are gaining increasing attention in the HCI community [28, 42, 56]. Important types include wearable robotic limbs [59, 68], prosthetic limbs [26, 50], robots that roam on the user's body [17, 30, 60], and exoskeletons [22, 39, 44]. Actuating the user's body, exoskeletons integrate particularly close with the human body. The resulting embodied partnership opens up a unique potential for designing novel, embodied human-robot interactions [39]. However, to date, research on exoskeletons in HCI is rare and how to design for intertwined human–computer integrations is not well understood [43]. One of the few works on exoskeletons within the HCI community is Dexmo, an inexpensive and lightweight hand-based exoskeleton, used for motion capturing and force feedback in VR [22]. DigituSync, another prominent example, employs a dual hand-based exoskeleton to share hand

[1]We refer to this target group as *novice roboticists* for the remainder of the paper.

[2]https://github.com/HCI-Lab-Saarland/ExoKit

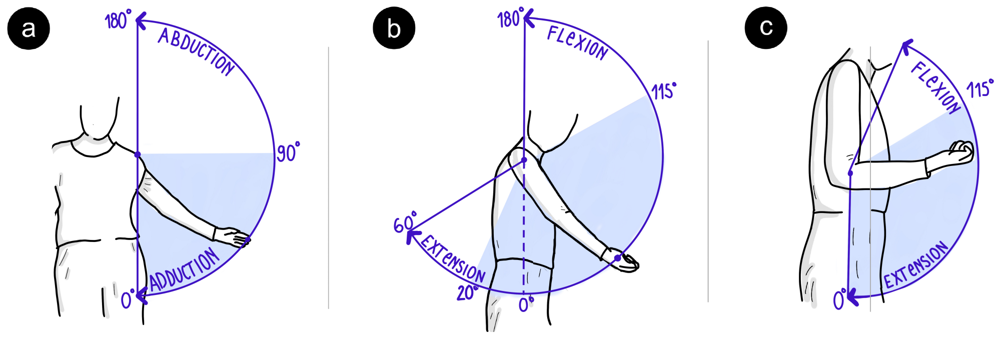

Figure 2: Degrees of freedom supported by ExoKit and the joints' ranges of motion (RoM) according to [46]. The colored areas indicate the RoMs supported by ExoKit.

movements between two people for skill learning through interconnected links, showcasing the potential of exoskeletons for novel and creative applications in HCI [45]. Beyond hand-based exoskeletons, only little research investigated the elbow [27, 64]. For instance, Homola et al. explored exoskeletons to aid game-based rehabilitation, leveraging a low-fidelity elbow-based exoskeleton built from Lego [27].

Exoskeletons have been extensively studied in the robotics community, but contrary to the HCI community, their primary focus lies on sophisticated mechanical designs whose objectives are stability, force, the support of complex joint types, and the prevention of misalignments with the human body, along dedicated control strategies [25]. Such exoskeletons are often developed for specific applications [25], such as industry [10], military [54], rehabilitation [53], and daily life assistance [29, 31], rather than as a generic design that allows for a broader exploration of interactions and rapid prototypability.

Contrasting the two communities, it is apparent that HCI's requirements for exoskeleton hardware inherently differ from those in robotics research. While the former focuses on user-centered design objectives, requiring rather low-fidelity prototypes that facilitate the exploration of novel human-exoskeleton interactions, the latter primarily excels in the development of sophisticated mechanical and control aspects. Attempting the needs of educators and tinkerers, EduExo[3] provides a valuable platform to teach fundamentals of exoskeleton principles to novices [2, 6]. However, it intentionally engages users with low-level motion control and provides limited actuated DoFs, and thereof still leaves a barrier to novices interested in human-exoskeleton interaction design from a higher-level perspective. Hence, there is a need for a toolkit that provides a low-fidelity exoskeleton prototype which empowers users to easily and rapidly customize the exoskeleton's design and interactive behavior to their needs, without having to handle the details of mechanical design and low-level motion control. We deliberately focused our toolkit on the shoulder and elbow to complement prior work in HCI which has primarily focused on hand-based and, to a lesser extent, on elbow-based exoskeletons [22, 39, 45, 64].

Biomechanics of the upper limbs. Considering the biomechanics of the human upper limbs is crucial for the design of arm-based exoskeletons. Key properties that are particularly relevant are the joints' degrees of freedom (DoFs), range of motion (RoM), and torque settings. The shoulder joint possesses three DoFs, enabling abduction-adduction (Figure 2a), flexion-extension (Figure 2b), and medial-lateral rotation. The elbow has two DoFs, allowing for flexion-extension (Figure 2c) and supination-pronation, which is the rotation of the forearm. The RoM defines the maximum arc through which the joint can move. Figure 2 provides an overview of the DoFs and RoMs supported by ExoKit, which lay within the typical RoMs of an adult's joints [46]. Another critical factor in the exoskeleton design and selection of suitable actuators is the torque required for the application, as different applications may require different torques; for example, rehabilitation exoskeletons may require less torque than industrial exoskeletons designed to support heavy loads. Furthermore, a user's torque perception is also influenced by various other factors such as movement direction or speed [34]. Therefore, the required torque settings need to be carefully determined for the specific use case.

2.2 Robot Programming

End-user robot programming focuses on enabling non-experts to control robot operations, broadening their deployment in real-world applications. To simplify robot programming, various approaches exist, such as API-based manual programming [3, 18], visual block-based programming [67], or programming by demonstration [1]. One interesting API-based approach is the Robotics API [3], an object-oriented framework for developing complex robotic applications. It abstracts key robotic actions and sensors as object proxies, which users can sequentialize and parallelize in an event-driven architecture. Such a "Trigger-Action Programming" approach has been widely used in HCI, too, enabling non-experts to program devices like smart homes [66] and humanoid robots [37] with relative ease. One crucial consideration in programming robots in this way is to identify essential functional abstractions which can be combined according to the tasks the robot should perform [16]. Striking a balance between exposing low-level functions for flexibility and providing high-level abstractions for ease of use is critical. While prior work has investigated this in domains like industrial or social robotics [3, 37], no such abstractions have been established for exoskeletons to the best of our knowledge. We argue that

[3]EduExo Kit: https://www.eduexo.com/eduexo-kit/, last accessed July 3, 2024

identifying a suitable level of functional abstractions for novice roboticists is one crucial aspect to facilitate rapid prototyping of human-exoskeleton interactions, empowering them to focus on high-level augmentation strategies rather than low-level motion control.

2.3 Robotics Toolkit Research in HCI

Toolkit research plays an important role in HCI by making new technologies accessible to broader audiences, reducing complexity, encouraging creative exploration, and integrating with existing practices [35]. HCI toolkits typically achieve these goals by encapsulating low-level domain knowledge, lowering the barrier to entry for non-experts (e.g., [4, 36, 38]). However, despite being established in HCI, toolkits specifically addressing robot prototyping remain limited [62]. A few notable examples include Blossom [62], Phybots [32], and WRLKit [56]. Suguitan et al. identify three key requirements for their robotics toolkit, two of which are particularly relevant to ExoKit: accessibility for rapid prototyping and flexibility to customize both the robot's design and behavior [62]. Similarly, Phybots is a toolkit targeted at researchers and interaction designers to rapidly prototype with locomotive robots through hardware and a software API [32]. WRLKit enables designers to prototype personalized wearable robotic limbs tailored to specific tasks and body locations through a computational design tool, thereby lowering the entry barrier to wearable robotics for the HCI community [56]. Inspired by these approaches, we address the gap in wearable robotics toolkits in HCI with ExoKit, a do-it-yourself toolkit which lowers the entry barrier for prototyping human-exoskeleton interactions.

3 Design Considerations

With ExoKit, we set out to develop an exoskeleton specifically for the arms with up to 3 active degrees of freedom (DoF) on each arm. We extend beyond existing HCI exoskeleton prototypes, which have largely focused on the hands and elbows to date (e.g., [22, 27, 45, 64]). By targeting the arms including the shoulder joint, we aim to open up a wider range of applications that involve larger body movements, such as motion guidance in sports, physical rehabilitation, and experiences in VR. ExoKit's objective is to facilitate the prototyping of human-exoskeleton interactions for novice roboticists in the early design stages. Its design considerations are guided by the five goals of HCI toolkit research [35] and informed by literature on wearables [20, 56] and exoskeletons [58, 61]:

Ease of development. Programming the interactive behavior of robots is a challenging task for novices [1]. To ease development and empower designers to focus early on on the prototyping of novel interactions, ExoKit should

- provide *powerful functional abstractions* that offer relevant functionalities for programming versatile interactive behaviors in few lines of code. In line with principles of end-user robot programming (e.g., [16]), these should abstract away the details of low-level motion control and instead allow the developer to focus on higher-level augmentation strategies,
- enable *interactive reactions from sensing to actuation.* As a body-centred technology, it is essential to provide functionalities that sense and interpret body motions in real-time, such as a limb's motion direction, position, or speed. This allows designers to employ these as conditional triggers for specific exoskeleton actuation, creating complex interactive behaviors.
- should provide an infrastructure that allows to *experience the implemented interactions with the prototype in real-time*, thus enabling designers to iteratively improve the design – a key principle in interaction design [52].

Fabrication. In line with design principles discussed in other physical toolkit papers in HCI, ExoKit should

- use *accessible materials and fabrication processes.* To empower new audiences and integrate with existing practices [35], we intend to use technologies widely used in HCI, design and making (e.g., 3D printing, Arduino) for building and implementing the prototype. This allows the community to use, extend and tailor the toolkit to their needs.
- consist of *modular components* that facilitate easy assembly of the prototype and rapid exchange of functional components after fabrication. This aligns with the iterative nature of interaction design and prior approaches of robot toolkits in HCI [12, 13].
- be *customizable to individual body characteristics*, an essential consideration in wearable robotic toolkits [56]. As different body characteristics can affect an exoskeleton's fit, function and wearability, ExoKit must offer mechanisms to easily adjust the prototype dimensions to different arm lengths and circumferences.

Safety. ExoKit must involve *safety measures*, which are essential for physical HRI [16, 56]. Actuating the human body, we must provide mechanisms that ensure that the applied forces and ranges of motions are safe and comfortable [61], even in case of inexperienced developers. This also involves, for instance, to accommodate motors with different maximum torques to let designers decide what is a safe range required for their application.

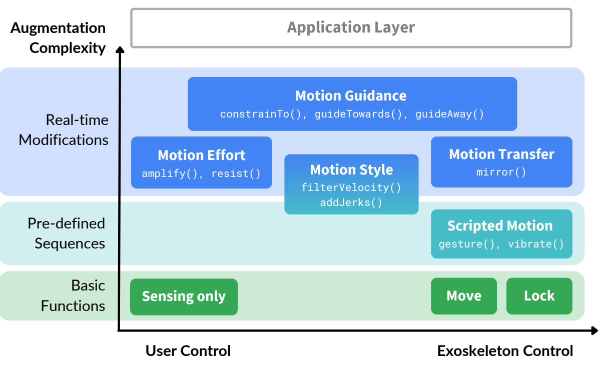

Figure 3: ExoKit's key functionalities comprise basic functions and five augmentation strategies. They are conceptually organized in a two-dimensional space: Control determines whether the user is in full control of movement, is guided or constrained, or fully controlled by the exoskeleton. Augmentation complexity captures the degree of abstraction, from basic functions up to real-time modification of user movement.

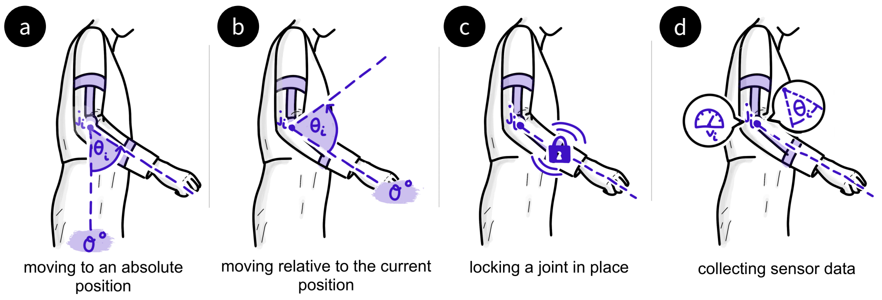

Figure 4: Basic functions offered by ExoKit: Moving a joint j_i to an angle θ_i expressed as (a) absolute w.r.t. j_i's calibrated zero-degree angle, or (b) relative to the joint's current angle; (c) locking joint j_i in place, or (d) collecting real-time sensor data of selected joints.

4 ExoKit's Interaction Concepts

ExoKit's objective is to facilitate the design of interactions for arm-based exoskeletons. For this purpose, it provides functional abstractions, which designers can combine into meaningful interactions. Beyond basic sensing and actuation, we identified five essential clusters of motion augmentation strategies that are recurrently used in the state-of-the-art exoskeleton literature [7, 14, 15, 25, 53]. We have organised them in a conceptual space, which is defined by two key dimensions of motion augmentation (see Figure 3): *control* and augmentation *complexity*. *Control* captures the extent of control the user vs. the exoskeleton have over the movement. At one end of the spectrum, the user is in full control and therefore can move freely to any target position. As we move along this axis, the user's freedom is progressively reduced while the exoskeleton is taking over more control, up to a point where the exoskeleton fully controls the user's motion. Augmentation *complexity* starts from basic functions, like sensing and moving, to pre-defined sequences of basic functions, up to augmentation strategies that modify a user's motion in real-time through a feedback loop.

In the following, we start with presenting the basic functions, followed by the higher-level motion augmentation strategies:

4.1 Basic Functions

Exoskeletons built with ExoKit are composed of either one or two arms. Each arm consists of up to three joints $j_i, i \in \{1, 2, 3\}$, which each can be either passive, sensing, or actuated. As a prerequisite for the following actuation functionalities, the designer must register the exoskeleton's configuration and calibrate the absolute zero-degree position in software for each actuated or sensing joint j_i once at system startup.

Actuation. An actuated joint j_i can be moved to a certain position. As body-centered technology, exoskeleton motions can be defined through angles θ_i in the user's joint angle space. Here, we distinguish between angles that are absolute with respect to the calibrated zero-degree angle of joint j_i, and relative angles which describe an angle relative to the user's current position (cf., Figure 4a, b). ExoKit enables designers to move a joint j_i (*moveTo()*) by specifying an absolute or relative target angle θ_i, the range ϵ around θ_i (useful for approaching an area centered at θ_i) and the velocity v. To fully prevent any user motion, a joint j_i can also be locked (*lock()*). This is achieved by activating the torque of the motor. Internally, the motor then maintains the position by continuously applying counterforces to the user's motion, thereby providing a locking experience for the user (Figure 4c). For ease of programming, all functions are provided for controlling an individual joint only, a set of joints, or the entire arm at once.

Sensing. ExoKit provides real-time sensor data about the exoskeleton's physical angle configuration, motion velocity, acceleration, and applied torques (Figure 4d). ExoKit can continuously stream this sensor data from selected or all joints through a serial port connection, making the data easily accessible to external applications. Secondly, ExoKit allows to use sensor data as conditional triggers in the firmware library: Developers can configure basic threshold- and range-based conditions about the exoskeleton's physical configuration (speed, acceleration, angle and torque). For instance, these help to detecting when the user moves outside a specified range of motion or exceeds a certain speed. Further, designers can access more complex movement states, which provide insights into each joint's motion direction and help to assess if the user's arm is at a desired angle configuration (or *pose*).

4.2 Scripted Motions

Building on the basic functions, designers can create and playback scripted sequences of motion, which are defined as a time series of basic move commands. Scripted body motions can be helpful to realize a specific repetitive motion with fixed parameters, or to play back motion sequences that serve as gestures (e.g., waving) and kinesthetic notifications [11]. For illustrative purposes, we provide functions for executing a waving gesture with the arm (*gesture()*) (see Figure 5a), and for generating a vibrating sensation through rapid back-and-forth movements of adjustable amplitude and frequency (*vibrate()*). The latter serves as a haptic notification for the user which can be delivered either to a single joint, a set of joints or the full arm.

4.3 Motion Transfer

The concept of motion transfer involves transferring human motion in real-time from one joint to another joint, from one arm to another arm, or even from one user to another person. This offers flexible means for applications: These involve guiding another

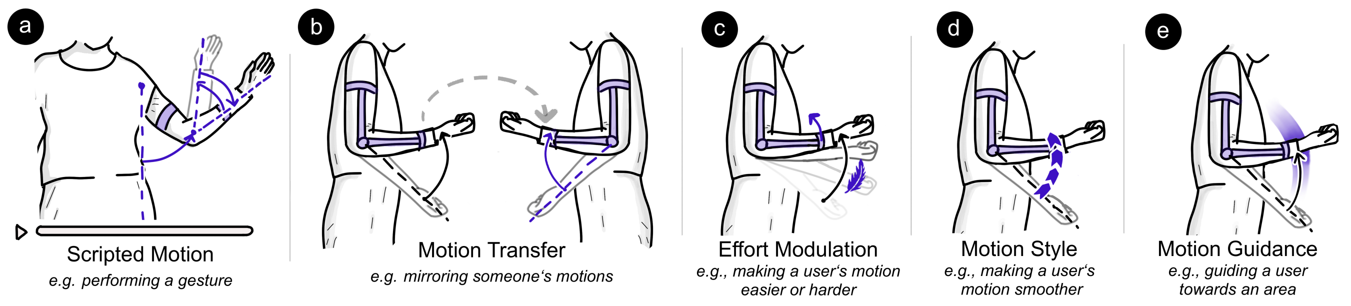
a
Scripted Motion
e.g. performing a gesture
b
Motion Transfer
e.g. mirroring someone's motions
c
Effort Modulation
e.g., making a user's motion easier or harder
d
Motion Style
e.g., making a user's motion smoother
e
Motion Guidance
e.g., guiding a user towards an area

Figure 5: ExoKit provides functional abstractions for five relevant classes of augmentation strategies which enable to (a) replay a scripted motion on demand, (b) continuously transfer one user's motions onto another one, (c) augment a user's motion effort by amplifying or resisting her ongoing motions, (d) alter the user's motion style, or (e) guide the user towards or away from an area.

user's motion, such as for motion teaching and skill transfer [41, 45] but also for creating shared kinesthetic experiences between users, or for programmatically remapping body input to output, such as mirroring an arm's movement to the other arm [51] or leveraging unemployed degrees of freedom for bodily control. ExoKit offers functions that facilitate motion transfer between two joints or exoskeleton instances, where in every time interval, the state of the sensing joints is read and the mapped actuated joints are moved accordingly. Designers can mirror the input signal from a sensing joint onto an actuated counterpart (*mirror()*) (see Figure 5b) and optionally scale the motion up or down by a user-defined factor f, while the system internally ensures that the scaled motion does not exceed the user's calibrated limits of their range of motion.

4.4 Modulating the Motion Effort

Modulating the motion effort comprises interaction strategies in which the exoskeleton applies torques τ in or opposite to a user's inherent motion, while preserving the user's ability to freely navigate in space. The resulting feeling of physical assistance or resistance can be leveraged by designers to modulate how users perceive their own motion, with stronger torques τ resulting in a stronger effect. Examples where effort modulation can prove beneficial are rehabilitation, where assistance eases a patient's movements while resistance enhances the muscle strengthening [14, 53], or virtual reality, where the immersiveness can be enhanced by adjusting the level of effort to better match the virtual environment [64]. ExoKit provides two pre-implemented functions which designers can use to increase (*resist()*) or decrease (*amplify()*) the motion effort, respectively, selectively for individual joints or applying to the entire exoskeleton. These functions include parameters that allow designers to adjust the assistive or resistive torque τ and to specify whether the strategy should be continuously active or only triggered when the user moves in a certain direction (see Figure 5c).

4.5 Modulating the Motion Style

Modulating the motion style refers to interaction strategies that modify the observable characteristics of a user's movement, such as speed or path, while keeping the user in control of the overall motion. These style modifications can range from subtle adjustments to more pronounced changes. Examples of motion style modulation from the literature include exoskeletons that suppress tremor [40], or exoskeletons that add perturbations to a user's movement, challenging their motion control [14]. ExoKit provides two pre-implemented functions to facilitate exploring modulations of motion style: (1) The exoskeleton tries to keep a user's motion speed within a pre-defined range (*filterVelocity()*), by applying assistive torques τ if they are too slow or resisting ones if they are moving too fast. Designers can fine-tune these effects by setting the velocity range and the forces applied to maintain it. (2) As an example of an artistic style modulation, the exoskeleton can introduce jerks (*addJerks()*) to augment a user's motion path for a more technical or robot-like motion style (see Figure 5d). A jerk consists of a short randomized movement that creates a distortion in the user's movement. Designers can adjust this effect by specifying the minimum and maximum angular displacements, the time intervals between jerks, and the number of jerks.

4.6 Motion Guidance

Haptic guidance comprises strategies in which the exoskeleton applies forces with the goal of guiding a user towards a certain point or area. These strategies are situated in-between full user control and full exoskeleton, where both effectively collaborate toward a shared goal. Depending on how parameters are adjusted, the designer can trade-off user control vs. exoskeleton control: exoskeleton control increases as guiding forces increase and as the range of motion becomes more constrained. Examples of haptic guidance include tunneling approaches in rehabilitation, where the patient's movement is locked within a predefined path or tunnel, and force fields that guide them back toward the center of it [14, 24, 53]. Beyond rehabilitation, they also hold promise in other areas, such as gaming and VR. ExoKit provides three complementary functions (see Figure 6): (1) The exoskeleton can constrain the range of motion of a joint j_i to an area around an absolute angle θ_i with range ϵ (*constrainTo()*). As soon as the user attempts to move beyond this range, the exoskeleton tries to move the user back to the boundary with maximum torque, thereby keeping the joint within the specified limits. (2) The exoskeleton can guide the user towards an area centered around θ_i with range ϵ (*guideTowards()*). Here, the exoskeleton applies assistive torques τ to j_i when moving towards the desired area and resists otherwise. (3) Lastly, the exoskeleton

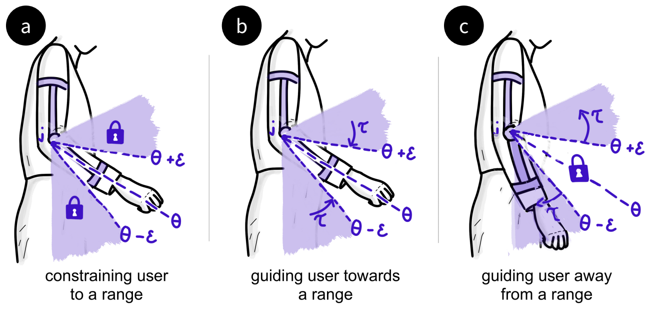
a
b
c
j
θ +ε
θ
θ -ε
τ
constraining user to a range
guiding user towards a range
guiding user away from a range

Figure 6: ExoKit's software features three functions for motion guidance: (a) constraining a user's motion to an area centered around θ_i with range ϵ, (b) guiding a user towards this area, (c) or guiding the user away from it.

can keep a joint j_i away from the area around θ_i (*guideAway()*), applying resistance with torque τ if the user is approaching the area and assistive torques otherwise. In addition to defining angle θ_i and range ϵ, designers can also specify the magnitude of the amplifying and resisting forces.

4.7 Programming Interfaces

ExoKit provides different interfaces to program the interactions, illustrated in Figure 7. Each interface is targeted at different experience levels, a recommended practice for end-user robot programming [1]. First, for novices in the early exploration stages, a simple command-line interface (CLI) can be used for executing the basic functions and higher-level augmentation strategies. It abstracts away complex programming concepts such as conditions or loops. We also offer a graphical user interface (GUI) programmed with Processing, which can be used in place of the CLI. The GUI allows novices to execute basic functions at the touch of a button, and tune parameters with sliders. Second, users can leverage a Processing library to define sequences of functions and strategies, enabling the programming of slightly more complex interactive behaviors. Third, more experienced users can program complex behavior within the Arduino firmware. The firmware builds on trigger-action programming, realized through an object-oriented event-driven architecture, in which users can combine basic functions and higher-level augmentation strategies into their own customized interactions. Similar to other works on end-user robot programming (e.g., [3, 23, 66]), users can define elements of parallelization, sequences, nested functions, or leverage incoming sensor data as conditional triggers. For more implementation details, see Appendix A and the supplemental material.

5 ExoKit Hardware Components

A key challenge in designing hardware components for an exoskeleton toolkit lies in balancing functional requirements, such as required degrees of freedom, stability, and safety, with designer's needs for rapid prototyping. Particularly in the early stages of interaction design, an important key characteristic for design is iteration [52], in which a designer usually explores a wide range of possibilities before converging on the most promising solutions.

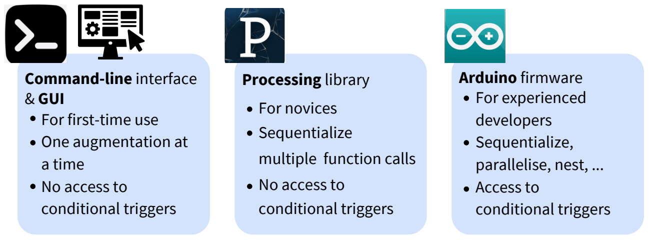
Command-line interface & **GUI**
• For first-time use
• One augmentation at a time
• No access to conditional triggers
Processing library
• For novices
• Sequentialize multiple function calls
• No access to conditional triggers
Arduino firmware
• For experienced developers
• Sequentialize, parallelise, nest, ...
• Access to conditional triggers

Figure 7: ExoKit provides a command-line interface and GUI, a Processing library, and an Arduino firmware library to program human-exoskeleton interactions.

While modularity, hands-on reconfigurability, and prototypability are well-established principles in HCI robotic toolkits (e.g., [12, 13]), these considerations are less prevalent in exoskeleton research so far. Traditionally, research on exoskeletons prioritizes functional requirements for high-fidelity solutions for specific applications, driven by the unique demands of each use case (e.g., [58]). We therefore note a disciplinary gap between the iterative and exploratory prototyping approach required in HCI and the function-focused demands of professional, specialised exoskeletons.

Building on the design considerations from Section 3, the design of ExoKit 's hardware components is centered on three aspects that integrate exoskeleton-specific requirements [58, 61] with HCI principles: (1) Modular components that facilitate hands-on reconfigurability of the exoskeleton's functionality, involving their DoFs, sensing and actuation capabilities, (2) alignment strategies that facilitate adjusting the exoskeleton's fitting for wearability and rapid prototyping, and (3) safety mechanisms suited for inexperienced developers.

All components offered by ExoKit can be fabricated using commercially available 3D printers and ABS filament, addressing the trade-off between stability and accessible fabrication. The toolkit further relies on widely-used off-the-shelf components, notably Arduino and Dynamixel motors. The current ExoKit prototype supports all motors of the Dynamixel XM series, providing forces up to 10 Nm with the strongest motor, and a maximum speed of 77 RPM with the fastest. The motors can be powered from 12 Volts using the supplied Dynamixel power supply or a wearable power bank. For torque sensing, the toolkit achieves rates of 80 Hz. ExoKit allows to configure exoskeletons with up to 6 active degrees of freedom – 3 per arm. The supported ranges of motion are 0 to 115 degrees for elbow flexion-extension, −20 to 115 degrees for shoulder flexion-extension, and 0 to 90 degrees for shoulder abduction-adduction, comparable to existing exoskeleton designs (cf., [9, 33, 55]). More implementation details can be found in Appendix B.

In the following, we detail on the individual hardware components offered in ExoKit and their roles in enabling reconfigurable, size-adjustable and safe exoskeletons:

5.1 Modular Components To (Re)Configure the Exoskeleton's Functionality

In early stages of prototyping, designers might not yet have a clear idea of the exoskeleton's final functional requirements. Thus, it is essential to create exoskeletons in HCI that support iterative

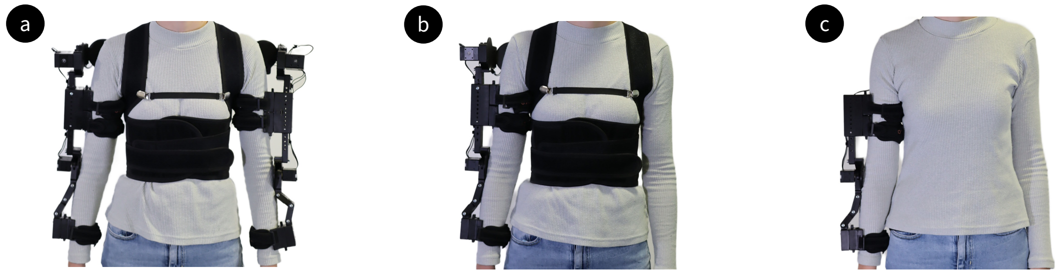

Figure 8: ExoKit's modular components allow to freely configure the degrees-of-freedom (DoF). Some example configurations include: (a) 6 DoFs at both arms, (b) 3 DoFs at one arm, (c) 1 DoF at one elbow.

development by empowering designers to add, remove, and modify components as requirements and the understanding of the problem space evolve. To address this, we propose a principled, modular architecture that is based on three key considerations: (1) In-line with modular exoskeleton designs [61], the exoskeleton should allow designers to adjust the DoFs to suit different application scenarios. (2) As exoskeletons serve a range of purposes—from motion capture [22] to delivering haptic feedback [64], or applying stronger actuation forces [19]—designers must be able to swap between sensing and actuating modules of varying strengths to match specific task requirements. (3) Inspired by exoskeletons where sensors extend beyond position sensing in joints (e.g., load cells [25]), the architecture must be able to accommodate additional sensing elements, ensuring that the exoskeleton can be adapted to changing requirements. We now discuss each consideration:

Customizable degrees-of-freedom. Different applications require varying DoFs in exoskeletons. The challenge for an exoskeleton toolkit with customizable DoFs lies in providing the necessary flexibility by freely configuring the DoFs after fabrication, without adding mechanical complexity, more components or reducing user comfort. Moreover, the provided components, even for more complex joints such as the shoulders, should be fabricable using accessible fabrication methods while providing a sufficient degree of stability. The toolkit extends beyond current exoskeleton approaches in HCI by offering support for the elbow and the more complex shoulder joints. It supports shoulder flexion/extension and abduction/adduction through two DoFs which can be actuated and a passive DoF that enables the exoskeleton to adapt to complex movements of the shoulder joint. In addition, the elbow is supported with one DoF in flexion-extension. The designer can easily adjust the hardware prototype by detaching joints to remove DoFs not needed for a specific application. For instance, designers can construct exoskeletons that support both arms, only one arm or the elbow (see Figure 8).

Modular joints. As exoskeletons can be either actuated, act as a sensing device, or be passive, it is crucial for designers to have the flexibility to choose the functionality for their application themselves. To enable hands-on reconfigurability, ExoKit offers three types of interchangeable joint modules: (1) *Actuated joints* comprise an actuator. A joint base, depicted in Figure 9a, serves for mounting a motor (Figure 9b). The base is designed to be compatible with all motors of the Dynamixel XM series. These motors inherently offer sensing capabilities (position, velocity). If desired, developers can modify the design of the joint base to include other motors with different dimensions. (2) *Sensing-only joints* measure the exoskeleton angle configuration, through which further motion properties, including velocity and acceleration, can be inferred. (3) *Passive joints* simply follow the user's motions, realizing a passive DoF. To realize sensing-only and passive joints, ExoKit provides a module that can be mounted on top of the joint base instead of a motor (Figure 9c). This module offers a space for an optional position encoder, hence can be repurposed as a sensing-only joint when the position encoder is attached (Figure 9d).

Modular links. Exoskeleton links are structural components that connect and transmit forces between joints. In ExoKit, we provide two types of 3D-printable passive links. The first is a simple rigid rod, used to stably connect the shoulder and elbow joints. The second is a flexible link made of 3d printed chain segments informed by the mechanism presented by Tiseni et al. [65]. The advantage of this approach is that the mechanism remains rigid in the direction where torque is applied while complying with user's complex body movements in other directions. In ExoKit, we employ this mechanism as a flexible link between the two shoulder joints, which helps to better comply with the shoulder's additional degrees of freedom, and as a link that connects the elbow joint with the lower arm cuff. If desired, both the rigid and chain links can be equipped with an additional load cell that captures the torques applied to the joints. As this is a commonly used control input in exoskeletons [25], this empowers designers to quickly integrate the load cell into their design without having to manually redesign the hardware. The Arduino firmware provides complementary support, allowing users to register the load cell as an exoskeleton component, use the sensing function to stream load cell measures to external applications and deploy them as runtime triggers in the event-based architecture. The link modules are depicted in Figure 10a and 10b.

5.2 Modular Components to Support Size-Adjustability

Adjusting the exoskeleton to the user's body size is essential to prevent misalignments that could cause discomfort or hinder proper functionality through undesired torques [25]. There are two promising approaches to achieve better joint alignments: fabricating a personalized exoskeleton or integrating size-adjustable mechanisms

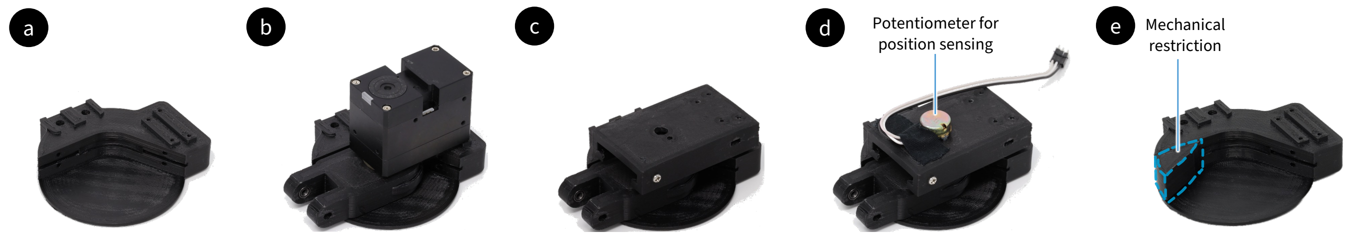

Figure 9: ExoKit features interchangeable joints based on a versatile joint base (a). This base can be fitted with (b) a motor for actuated joints, (c) a passive module to replace the motor, or (d) a passive module with an integrated position encoder for sensing-only joints. (e) Designers can insert mechanical restrictions to the base joint to further limit the offered range of motion by 15, 30 or 45° on each side.

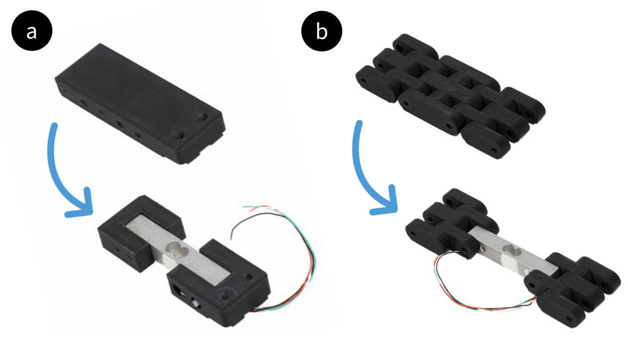

Figure 10: (a) Rigid and (b) flexible links without and with integrated load cells, respectively.

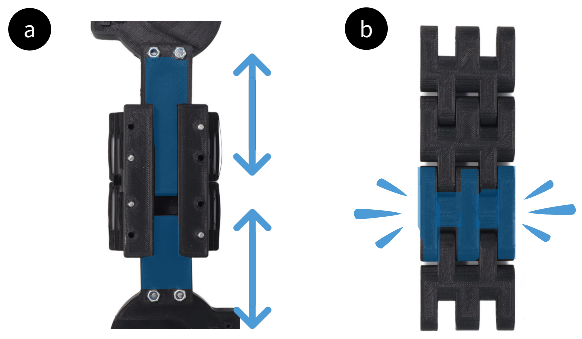

Figure 11: (a) The rigid joint can be adjusted in length through a rail system which allows to move the 3d printed rods further in and out. (b) The flexible link can be easily adjusted in size by adding or removing chain segments.

post-fabrication [58]. ExoKit follows the latter, enabling hands-on customization which avoid the need to reprint new parts for every user, thereby facilitating rapid prototyping in collaborative settings. All of ExoKit's links are designed for size adjustability. In line with strategies presented in exoskeleton literature [5, 70], the rigid links on the upper arm and back use adjustable rail mechanisms to accommodate a range of body sizes within the 95th percentile [21] (see Figure 11a). The chain link introduced in Subsection 5.1 can be adjusted to different shoulder circumferences and lower arm lengths by simply attaching or detaching chain segments (see Figure 11b). Finally, the arm cuffs which connect the links to the human arm come in two different sizes, and can be stacked together to provide a larger area that holds the exoskeleton in place.

5.3 Physical Safety

Safety is an important concern in exoskeletons, particularly if large forces are involved. Based on considerations in robotics literature (e.g., [61, 69]), ExoKit comprises three layers of safety, which cover mechanical, software, and user-centered safety measures, specifically targeted at novice developers:

The first layer presents mechanical safety measures to maintain a safe range of motion, an important safety goal in exoskeletons [61]. Designers can add 3d printed mechanical restrictions to the base joints (see Figure 9e). Through a simple plug-and-play mechanism, these mechanical restrictions physically prevent the motor from exceeding the defined range of motion. They are offered in 3 different sizes (restricting by 15, 30 or 45° on both sides) and can be added, removed, or exchanged after fabrication as needed.

The second layer realizes safety mechanisms in software, another essential building block of user safety in exoskeletons [61]. ExoKit implements a routine continuously monitoring whether the user's configured range of motion is exceeded and if so terminates and shuts down the system.

The third layer comprises user-initiated safety measures, to empower the user to actively stop the exoskeleton if required. ExoKit incorporates a panic button, a common feature in human-robot interaction [69], which the user can either hold in the hands or attach somewhere on their body. Analogously to the prior layer, if triggered, the system immediately terminates the program and shuts down. This can be a particularly helpful measure to ensure safety during initial development or debugging of some new functionality.

6 Validation

We follow Ledo et al.'s strategies for evaluating HCI toolkits [35]: (1) With implemented functional example applications, we demonstrate how ExoKit enables designers to easily customize exoskeleton's design and interactive behavior. (2) With our first usage study, which combines elements of walkthrough demonstrations and free

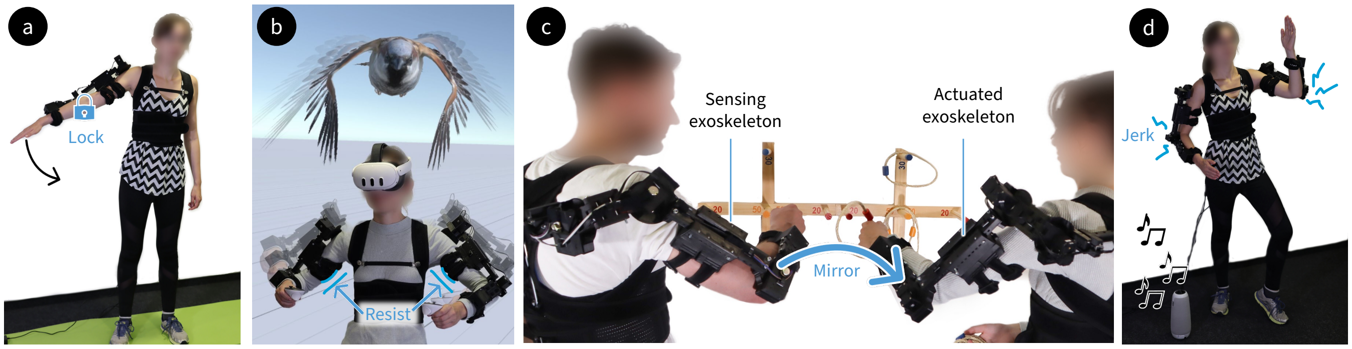

Figure 12: ExoKit can be used for versatile applications, such as for (a) physical motion guidance for strength exercises, (b) haptic feedback in VR, (c) collaborative body-actuated plays, and (d) artistic modulation of motion in real-time.

exploration, we gather insights in ExoKit's utility. (3) Finally, with a second usage study, in which participants built their own application with ExoKit, we gained insights into users' workflows and encountered challenges.

6.1 Example Applications

We illustrate the broad applicability of ExoKit with four implemented application examples. These are illustrated in Figure 12. The examples have been selected to cover different application areas, exoskeleton configurations, and augmentation strategies. All have been implemented with the Arduino firmware. We report on the first one in more detail to demonstrate the workflow.

Designing motion guidance for strength exercises. A promising area of exoskeletons is rehabilitation and home therapy. We used ExoKit to assist in the correct execution of an upper arm exercise aimed at strengthening muscles, inspired by an example provided in Physio@Home [63]. In this exercise, the user stretches their arm out sideways and moves it up and down repetitively, ensuring the arms don't leave the body plane. At the same time, the elbow must remain fully extended throughout the exercise. The exoskeleton's role is to help the user maintain proper form during execution (Figure 12a). We began by addressing how to keep the user's shoulder movement aligned within the body's side plane. To do this, we built an exoskeleton with an actuated joint at the shoulder (on the side) to modify sideway shoulder motions, along with a passive joint at the back. Initially, we used the *constrainTo* function to lock the user's movement within a narrow area inside the desired body plane. However, we noticed that this reduces the user's agency substantially. To give the user more control, we switched to the *guideTowards* function, which guides the user back toward the desired plane. Since the shoulder can exert significant forces during this motion, we leveraged a stronger Dynamixel motor (XM540-W270-T). This motor provides enough power to apply both assisting and resisting forces at the shoulder and we fine-tuned the torques to be effective yet gentle. Once satisfied with the shoulder motion, we integrated an actuated joint for the elbow, using the weakest Dynamixel motor (XM430-W210-T) and added a *lock* function to ensure that the elbow does not drift from the desired position as the user moves.

Kinesthetic feedback for avatar embodiment in VR. An important research area in VR is providing haptic feedback [22, 64]. We implemented a VR environment that uses ExoKit to create immersive kinesthetic feedback for embodying the motion of an avatar. In our application, a flying game, the user can control the motion of an avatar's wings, by moving one's arms, and feel the corresponding kinesthetic real-time feedback (see Figure 12b). For instance, when embodying a dragon, characterized by heavy, powerful movements, a designer can adjust the exoskeleton's 6 actuated DoFs with the *resist* function to increase resistance, simulating large, forceful wing beats. Conversely, for a sparrow, with its fast and light motions, the designer can reduce resistance or even employ the *amplify* feature to emphasize the sparrow's agility. Through iterative testing and adjustment of the *resist* and *amplify* parameters, we can fine-tune the kinesthetic feedback, creating avatar-specific haptic experiences.

Collaborative body-actuated play. Body-actuated play [47] leverages body-actuating technologies for novel kinds of creative, shared bodily experiences. Using ExoKit, we developed a collaborative game that demonstrates how designers can craft interactions between multiple players each wearing an exoskeleton. In this game, one player (P1) controls the body movements of another player (P2). The goal is to solve a physical color sorting game by making P2 sort as many rings correctly as time permits. P1 is wearing a 3-DoF exoskeleton arm with sensing capabilities that captures her movements. P2 wears a fully actuated 3-DoF exoskeleton that mirrors P1's motion in real time (see Figure 12c). This behavior was rapidly implemented with the toolkit's *mirror* function, which supports customizable body remapping strategies in few lines of code. To make the game more demanding for advanced players, we developed a second level that adjusted how the transferred motions are scaled by changing one parameter in the software. Finally, we implemented a third level, that goes beyond a direct 1:1 mapping of body parts and instead maps P1's sideways shoulder motions onto P2's elbow motions, and P1's elbow motions onto P2's shoulder joint to challenge coordination.

Enhanced artistic performances. Exoskeletons also offer interesting applications for artistic performances. Using ExoKit, we developed an arm exoskeleton that artistically modifies a dancer's movements in real time. The dancer defines and executes the overall motion of the arms, while the exoskeleton enhances the style of the motion to be more jerky and robot-like (see Figure 12d). By leveraging the pre-implemented *jerk* function, the designer can introduce variability in the dancer's movements, making them more abrupt. Designers can easily adjust jerk parameters, such as amplitude, frequency, and velocity, to suit the performance, from subtle modifications to exaggerated effects. Moreover, the toolkit allows designers to easily implement specific body gestures that serve as triggers for starting and stopping the motion augmentation. We implemented an "arm stretched out" gesture (elbow is at 0 degree angle) for triggering the motion style modulation, ensuring the augmentation aligns with the dancer's intent and preferences.

6.2 Usage Study 1: Experiencing ExoKit

The objective of the first study was to gain insights into ExoKit's utility, understand how it aligns with the key design requirements, and identify opportunities for improvement.

6.2.1 Method. To address this objective, we conducted a usage study with six participants in two workshop sessions. The workshop setting has been proven useful to evaluate toolkits in prior work (e.g., [36, 49, 57]). Each workshop combined walkthrough demonstrations (cf., [35]), where participants are presented with ExoKit's functionalities, with phases of free exploration and discussion.

Participants. The first workshop was conducted with three participants experienced in interaction design, representing our target group of novice roboticists: a postdoctoral researcher experienced in interaction design for VR/AR (ID1, 28, male) and two graduate students (ID2 & ID3, both 23, female and male) with research experience in interaction design for wearables and first hands-on experience with exoskeletons, respectively. The second session comprised three participants experienced in hardware and fabrication, serving as technical experts to provide feedback on the design and potential improvements: a postdoctoral researcher working on robotics and haptics (HF1, 32, male), a PhD student primarily experienced in fabrication (HF2, 24, female), and an undergraduate student with more than ten years of hardware tinkering experience (HF3, 31, male).

Procedure. After providing informed consent, we introduced the workshop participants to ExoKit, presenting its conceptual space (Figure 3) and physical hardware components (Section 5). Participants received an exoskeleton consisting of one fully actuated and one sensing arm, as well as a printed manual for all provided interface functions. In the first half of each workshop, they familiarized themselves with the functions offered by ExoKit while the exoskeleton was still attached to a mannequin (Figure 13a). In the second half, each participant was prompted to wear the prototype and encouraged to adjust its components to fit their body (Figure 13b,c). Taking turns, the other group members then used the command-line interface to trigger desired functions (Figure 13d,e). The workshop was interleaved with rounds of discussion and short brainstorming sessions. Sessions were audio- and video-recorded for later analysis. Each session took 3 hours.

6.2.2 Results. We collected findings based on our observations of participants' behaviors and statements. We group these results along key design considerations of ExoKit:

Explorative, creative and iterative design. Most importantly, in both workshops, the toolkit fostered a playful exploration of human-exoskeleton interactions, especially enabling rapid iterations on design parameters for various functionalities. Notably, the hardware group engaged deeply with extreme parameter settings to test the toolkit's limits, and quickly adopted a creative designer's perspective. For instance, they spontaneously crafted challenges for each other by leveraging motion transfer functions, where unconventional body remappings (such as mapping a shoulder movement onto the elbow) complicated the fulfillment of the challenge (Figure 13d). HF2 later highlighted how this could bear potential to design fun collaborative and creativity-based interactions between users. Similarly, in the interaction group, the exploration of resistance and amplification, and how these affected the body movements, immediately sparked ideas for possible applications, including a simulated weight-lifting in VR or leveraging them for avatar embodiment.

In a follow-up brainstorming session, participants proposed more ideas across various domains: Ideas in healthcare included providing adaptive support for weak arms (ID2) and mirroring motions between therapists and patients to aid rehabilitation (HF2). ID3 envisioned teleoperation scenarios, such as controlling a fatigued arm with the opposite limb during physically demanding tasks like wall painting. ID2 and ID3 also discussed about social applications, like providing hugs over a distance, enhanced with thermal feedback. Ideas for leisure and gaming featured interactive game interfaces and leveraging the exoskeleton to teach dancing or other sports (HF2). Finally, participants also highlighted ExoKit's prospective applications for STEM education (ID2, ID3).

Programmability and functional abstractions. Participants found the provided functional abstractions generally easy to understand and quickly grasped the concepts. They further appreciated the offered parameters which allow to rapidly modify the strategies' effects, such as tuning the motion style with different jerk parameters, or experimenting with varying torques applied for the motion effort strategies. This underlines the toolkit's potential to empower its users to leverage complex exoskeleton functionalities encapsulated by the identified functional abstractions. Further, participants acknowledged that the provided functional abstractions provide a broad and generic support for various applications. HF3 indicated that for more specialized purposes, it would be advantageous if the designer can define their own augmentation strategies building upon the provided functionalities. This resonates with the different ways of programming offered by ExoKit (cf. Figure 7), which facilitate the customization of new interactive behaviors. Finally, both groups suggested that for an initial interactive exploration of the provided augmentation strategies, a graphical user interface that visualizes predefined input parameter ranges for different functions would be a promising extension[4].

[4]Of note, the GUI presented in subsection 4.7 was implemented in response to this feedback.

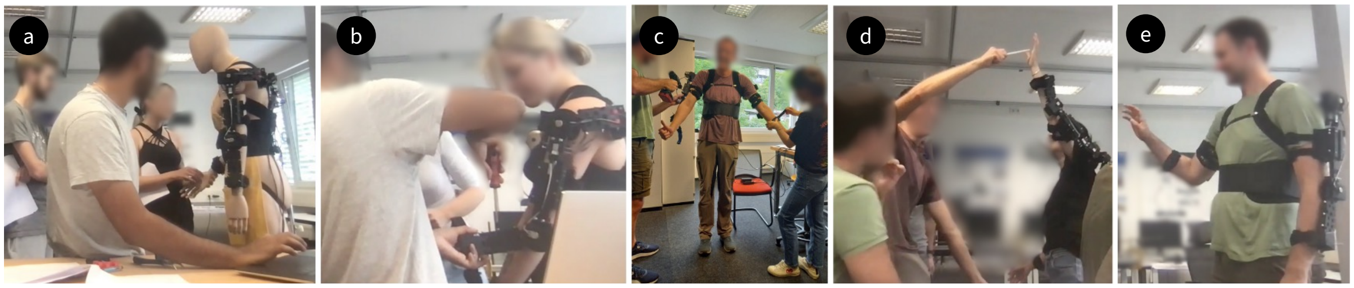

Figure 13: Workshop participants first familiarized themselves with the provided prototype (a), put it on and collaboratively adjusted the link sizes (b, c), and experienced the provided functions on their bodies (d, e).

Wearability. Both groups successfully adapted the exoskeleton's size by adjusting the link lengths to their body sizes, demonstrating how the modular components can enhance ergonomics. The interaction group expressed themselves particularly positive about the wearing comfort and lightweight design. Especially ID2, who was familiar with other open-source exoskeleton prototypes, acknowledged that the prototype built with ExoKit enhanced comfort. Finally, the hardware group offered suggestions to further improve wearability, as they criticized that attaching and detaching the prototype currently requires a second person. For instance, they suggested using more easily attachable backpack straps instead of the postural corrector, and sewing the joint elements directly onto some fabric to further enhance stability.

Customizability of hardware. To further customize ExoKit to the versatile needs of HCI researchers, both groups emphasized the importance of being able to integrate additional sensing and actuation mechanisms, such as thermal feedback or vibration. This requirement is already supported by ExoKit's current infrastructure which is intentionally built on Arduino and thereof naturally compatible with a wide set of commonly used, relevant electronics. Additionally, HF3, who has a strong background in hobby tinkering, requested support for low-cost joint modules as the currently supported Dynamixel motors are comparatively expensive for private use beyond research.

6.3 Usage Study 2: Prototyping with ExoKit

The goal of the second study was to gain rich qualitative insights into how users approach the toolkit to build an application, their workflows, the components they would use and encountered challenges.

6.3.1 Method. Inspired by Lei et al. [36], we conducted a usage study in which users engaged with ExoKit over a longer period of time.

Participants. We recruited three participants from the target group, all with an interest in learning about human-exoskeleton interactions: an undergraduate student with some theoretical knowledge of human-exoskeleton interaction who self-reported basic electronics and programming skills (P1, 22, male), a graduate student with some background in design and fabrication of actuated systems, good electronics and basic programming skills (P2, 28, male), and a PhD student in HCI interested in sports and accessibility without any prior robotics experience but basic electronics and good programming skills (P3, 26, male).

Procedure. Participants worked in a quiet space equipped with ExoKit (software, tools, user manual) and pre-fabricated hardware modules (Figure 14a). In response to the feedback from usage study 1, we additionally introduced a GUI to facilitate first-time exploration, replacing the command-line interface (cf., subsection 4.7). Participants were tasked with creating a functional human-exoskeleton interaction for an application of their choice, with no time constraints. The only requirement was to present a live demonstration at the end. After a brief introduction, we taught participants how to get the basic setup (comprising an Arduino, a connected motor and a programming interface) running. Next, participants familiarized themselves with the offered functions with the help of the GUI before starting to develop their own applications. To gather in-depth qualitative insights, we conducted observations, took notes, and engaged in short conversations with the participants from time to time to assess their progress. The study concluded with semi-structured interviews, which were audio-recorded and transcribed for analysis.

6.3.2 Results. We thematically analyzed our notes and interview transcripts and generated the following themes:

Implemented applications. P1 created a **sound synthesizer** where the position of the elbow joint is transformed into a sound in real-time (Figure 14b). Similar to a theremin, the farther the lower arm moves towards flexion, the higher the pitch becomes. The idea was inspired by the user manual which provides application ideas to inspire novel users, and resonated with P1's interests as he has *"some colleagues [who] built an application to use novel input devices to create sounds"* . P1 used the Arduino library to continuously stream motion data to an external application. He implemented a small Processing application to parse and sonify the incoming sensor data using Processing's sound library[5].

P2 designed a **device that teaches him to do an underhand toss**, as he was *"not particularly athletic and I thought that an exosuit could teach me to kind of perform this underhand toss"* (Figure 14c). Using two motors, P2 actuated the right shoulder and elbow to support flexion-extension. With the help of the Processing library,

[5]Processing's sound library: https://processing.org/reference/libraries/sound/index.html, last accessed December 3, 2024

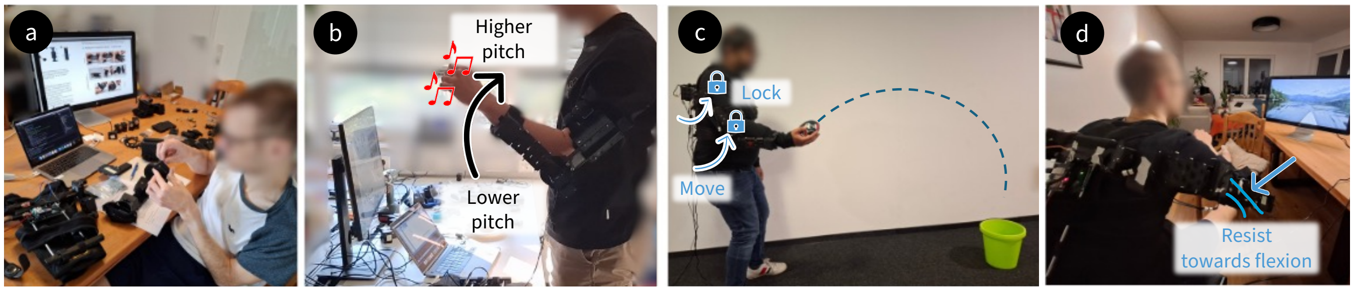

Figure 14: Participants used ExoKit's software and hardware components (a) to implement an application of their choice: P1 created a sound synthesizer (b), P2 an exoskeleton that teaches how to do an underhand toss (c), and P3 a rowing application (d).

ID	Application	Programming Interface & Functions	Hardware Components
P1	Sound synthesizer	Arduino library and Processing application; continuously streaming joint angle data (2.5 h)	1 active elbow joint module, links, cuffs (20 min)
P2	Teacher for learning a toss	Processing library; a sequence of joint and arm-wise *moveTo()* & *lock()* functions (3.5 h)	2 active (elbow & shoulder side) and 1 passive shoulder back joint module, links, cuffs, back rail & posture corrector (1.5 h)
P3	Rowing trainer	Arduino library; a sequence of *resist()* functions with variable torques, executed in parallel on both joints (40 min)	2 active elbow & 4 passive shoulder joint modules, links, cuffs, back rail & posture corrector (3 h)

Table 1: An overview of the applications built in the second usage study, including the software and hardware components that the participants decided to use. Within the indicated time, participants familiarized themselves with the library functions, implemented the software, and learned to assemble the exoskeleton prototype.

P2 implemented the required motion sequence: both actuated joints first move to a neutral position, then the exoskeleton performs a throwing motion by moving the joints towards the targeted angles, followed by a brief lock to keep this position.

P3 created a **training device for rowing**, using an exoskeleton with two actuated elbow joints to provide resistance during arm movements. P3 implemented a personalised training plan, in which the provided resistance increased in 5-second intervals to a user-defined maximum before gradually decreasing again. For a more realistic feeling, the resistance was only provided during elbow flexion. P1 synchronized his rowing movements with an online first-person rowing video for demonstration (Figure 14d).

Table 1 provides an overview of the software and hardware components that the participants used and the time they took to build the application.

Participants reflected on the learning curve, which could be expected given the multitude of modular software and hardware components with which the participants had to become familiar first. For instance, P3 stated: *"I would have a huge advantage if I do it again because I mean at the end it is all very clear and I understand the modularity [. . .] but it was just a lot to learn"*. Of note, in the end, all participants were able to design, build and implement their own exoskeleton interaction.

Different approaches to developing an application with ExoKit. Interestingly, the three participants demonstrated three distinct approaches to building applications with ExoKit: P3 started by designing the complete hardware, considering *"how large should the exoskeleton be, which joints should it involve, should it actually have all the joints and if yes should they be passive or not be passive"*, before programming the interaction in one go using the Arduino library. Contrary, P1 prioritized getting the programming logic right before assembling any hardware. Working only with a minimal hardware setup (a motor attached to the Arduino), he explained *"'because while the exo is not fully assembled, I found it easier to actually test the [functionalities] I wanted to use. Especially since I was not that familiar with ExoKit yet, I really enjoyed it was not assembled yet"*. P2, being less experienced in interaction design and programming, began by studying tutorials and his own body motions to identify the desired sequence, first drafting it on paper *"to translate this into sort of like a programmatic approach"* and building the corresponding hardware. Afterwards, he deployed an iterative programming approach, gradually increasing the complexity and refining parameters. These distinct approaches highlight how ExoKit's modularity provides flexibility to accommodate users with different levels of experience and preferences.

The libraries' perceived ease of use depends on user's prior experience. All participants highlighted that the GUI is *"most ideal for exploration"* (P1) as it requires minimal input: *"you just have to click, you don't have to enter numbers"* (P3). Moreover, the participants emphasized that their first application ideas were inspired during the initial exploration with the GUI, thereby demonstrating its effectiveness in supporting novice users. In contrast, we noted that the perceived ease of the Arduino library varied with participants' prior programming experiences, underscoring the importance of providing different programming interfaces. P3, with the most programming experience, found the Arduino library *"easy enough"* for his needs. P1, a less experienced student, mentioned that it was challenging in the beginning, but he eventually gained *"enough knowledge to work and play with it"*. For P2, we provided support after observing that he got stuck trying to learn the Arduino API, and pointed out the possibility of starting with a simpler programming interface first. P2 concluded that the Processing library aligned better with his skill level: *"If the [GUI] is for people who have no programming experience and if using the Arduino library is for people who have lots of development experience, then I should stick to my strengths which is being in the middle"*.

Helping users to help themselves. Our study showed that ExoKit supports users to adopt their own workflows, but also revealed opportunities to improve support when they struggle. First, similar to the workshop studies, the participants quickly grasped the provided functional abstractions and explored parameters independently with the GUI, demonstrating the suitability of the abstraction levels and the provided support through the manual. Second, to program the interactions, P1 and P3 successfully built on code snippets provided in the manual and tailored them to their needs. P1 and P2 aspired even more examples and detailed code explanations in the manual as this would further ease the learning process. Third, P1 and P2, who independently assembled the hardware, emphasized that *"the instruction manual made it pretty straightforward to figure out how it is to be assembled"* (P2), though P1 noted occasional confusion due to similar-looking parts. We observed similar moments of confusion for all participants, such as inserting the rigid link into the arm rail first in the wrong orientation. We see potential to reduce such mistakes by integrating more affordances and constraints into the 3D-printed components, mechanically preventing incorrect orientations or connections. Finally, participants recommended to further enhance the assembly instructions through videos (P3), more visuals (P2), or schematic drawings (P1).

6.4 Discussion, Limitations & Future Work

We now discuss lessons learnt from the implementation of the application cases along with insights from the usage studies. We also discuss limitations of our current implementation and derive implications for future work on exoskeleton toolkits in HCI.

Exploring human-exoskeleton interactions. Our example applications and usage studies demonstrate how ExoKit facilitates exoskeleton prototyping and design iteration for non-experts, and hence lowers the entry barrier for engaging with exoskeletons. While prototyping the application cases, we intensely leveraged the modular design of the exoskeleton, which allowed us to easily reconfigure prototypes by swapping electronics and functional parts. As demonstrated in the usage studies, the command-line interface and the GUI enabled rapidly exploring functions and iterating on their parameters. It became apparent that the developers needed to fine-tune the parameters of real-time modifications, which modify ongoing user motions, such that the combined motion of both user and exoskeleton felt smooth. Here, too, the provided interfaces proved helpful for rapid iterations. We also noted that the screws required for assembly slowed down prototyping. While other toolkits such as MiuraKit [13] deploy plug-and-play mechanisms which allow for rapid reconfigurability of the robotic structure, they must be carefully designed in case of ExoKit to withstand the mechanical stresses on the exoskeleton, a point for future refinement. Furthermore, ExoKit opened up new possibilities for exploring human-exoskeleton interactions, beyond the initially anticipated use cases. Both the study participants and the authors came up with novel ideas for interaction designs, such as unconventional body remapping via motion transfer functions. This underlines ExoKit's potential as a platform for creative exploration for the HCI community.

Ease of programming interactive behavior. Our usage studies showed that all users could rapidly grasp and modify the behavior of the augmentation strategies through the command-line interface and GUI. Our example applications further demonstrate more complex use cases which were implemented with the Arduino firmware. While the provided functional abstractions present a first essential step to making complex exoskeleton behaviors accessible to novice roboticists, we identified the following functionality to be desirable in future extensions: first, the possibility to control exoskeleton motion in 3D world coordinates, rather than only in joint angle space; second, adding more sophisticated force profiles for motion guidance [14, 53], and third, a record-and-replay mechanism for asynchronous motion transfer that captures motions and replays them at a later point [14, 53]. We further found that the perceived ease of use of the provided programming interfaces depends on to the user's programming experiences. Drawing on these insights, an additional worthwhile direction to further ease programmability and make the toolkit accessible for even broader audiences is to extend beyond text-based programming to novice-friendly visual programming, such as block-based approaches that could be used instead of the Arduino library. Finally, a simulation tool could provide further support to the designer, for simulating and testing interactions before deploying them on an exoskeleton. Yet, it is an open question how to most effectively simulate and communicate the effects of body-augmenting systems on the human body.

Extending support to other body parts. ExoKit currently supports up to three active DoFs for each arm, opening up new opportunities for interaction design in the HCI community. However, we are far from supporting the whole body. Extending support to other body parts could unlock new interaction opportunities. For instance, future iterations could add more DoF at the arms, by including internal rotation. Another promising avenues is to extend support to the lower limbs, including the hip and knees. We expect that our fundamental design decisions (modular components, exchangeable joint types, etc.) will generalize to the lower limbs. However, these pose additional safety challenges, as stronger actuation forces are required and if not done right, these could negatively affect the user's balance. In consequence, future work should also expand the conceptual space to address new augmentation strategies for the lower body, such as gait assistance for leg exoskeletons [8].

Fabrication & Cost. The cost of realizing an exoskeleton with ExoKit centrally depends on the cost of the motors (between 270 to 430 USD). In comparison, the cost of the fabrication materials is negligible. Relying on 3d printing filament is one factor that makes this technology accessible to a larger audience. While the resulting prototype is sufficient for practical use in early iterations, of note, it also inevitably involves compromises when compared to professional exoskeletons, which are often made from metal structures suited for bearing heavy loads. Finally, while the overall cost is only a fraction of the price tag of a professional exoskeleton, this may be still too expensive for private users. It is a worthwhile direction to investigate how lower-end, lower-cost motors can be integrated with ExoKit. We invite the community to contribute to the advancement of ExoKit, adapting existing modules and sharing new ones.

7 Conclusion

This paper presented ExoKit, a do-it-yourself toolkit that empowers novice roboticists with basic electronics and programming skills to rapidly prototype interactions for functional lo-fi exoskeletons targeted at the arms. ExoKit features modular hardware components that allow to easily reconfigure its active degrees of freedom, adjust component's dimensions to accommodate various body sizes, and safety mechanisms. We conceptually identified relevant high-level augmentation strategies and provide them as functional abstractions that simplify the programming of interactive behaviors. These functions are readily accessible and customizable through a command-line interface, GUI, Processing library, and Arduino firmware. Through application cases and two usage studies, we demonstrated ExoKit's potential to ease the development of human-exoskeleton interactions and support creative exploration and rapid iteration in early-stage interaction design. We hope that this work will inspire HCI researchers to explore the emerging field of human-exoskeleton interaction and unlock its potential for innovative applications.

Acknowledgments

We thank all participants of our usage studies and express our particular gratitude to Ata Otaran for his feedback. We also thank the reviewers for their valuable comments.

References

[1] Gopika Ajaykumar, Maureen Steele, and Chien-Ming Huang. 2021. A Survey on End-User Robot Programming. *Comput. Surveys* 54, 8, Article 164 (oct 2021), 36 pages. https://doi.org/10.1145/3466819

[2] Mohammed Al-Tashi, Bengt Lennartson, Max Ortiz-Catalan, and Fabian Just. 2024. Classroom-ready Open-source Educational Exoskeleton for Biomedical and Control Engineering. *at - Automatisierungstechnik* 72, 5 (2024), 460–475. https://doi.org/doi:10.1515/auto-2023-0208

[3] Andreas Angerer, Alwin Hoffmann, Andreas Schierl, Michael Vistein, and Wolfgang Reif. 2013. Robotics API: Object-oriented software development for industrial robots. *Journal of Software Engineering for Robotics* 4, 1 (2013), 1–22.

[4] Abul Al Arabi, Jiahao Li, Xiang 'Anthony Chen, and Jeeeun Kim. 2022. Mobiot: Augmenting Everyday Objects into Moving IoT Devices Using 3D Printed Attachments Generated by Demonstration. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 273, 14 pages. https://doi.org/10.1145/3491102.3517645

[5] Volker Bartenbach, Marcel Gort, and Robert Riener. 2016. Concept and Design of a Modular Lower Limb Exoskeleton. In *2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob)*. 649–654.

[6] Volker Bartenbach and Camila Shirota. 2020. Design and Impact of a Commercial Educational Robotic Exoskeleton. In *Inclusive Robotics for a Better Society: Selected Papers from INBOTS Conference 2018, 16-18 October, 2018* (Pisa, Italy). Springer, 136–140.

[7] Angelo Basteris, Sharon M Nijenhuis, Arno HA Stienen, Jaap H Buurke, Gerdienke B Prange, and Farshid Amirabdollahian. 2014. Training Modalities in Robot-mediated Upper Limb Rehabilitation in Stroke: A Framework for Classification Based on a Systematic Review. *Journal of neuroengineering and rehabilitation* 11 (2014), 1–15.

[8] Romain Baud, Ali Reza Manzoori, Auke Ijspeert, and Mohamed Bouri. 2021. Review of Control Strategies for Lower-limb Exoskeletons to Assist Gait. *Journal of NeuroEngineering and Rehabilitation* 18 (2021), 1–34.

[9] Pietro Bilancia and Giovanni Berselli. 2021. Conceptual Design and Virtual Prototyping of a Wearable Upper Limb Exoskeleton for Assisted Operations. *International Journal on Interactive Design and Manufacturing (IJIDeM)* 15 (10 2021). https://doi.org/10.1007/s12008-021-00779-9

[10] Robert Bogue. 2018. Exoskeletons–A Review of Industrial Applications. *Industrial Robot: An International Journal* 45, 5 (2018), 585–590.

[11] Adrian-Vasile Catană and Radu-Daniel Vatavu. 2023. Fingerhints: Understanding Users' Perceptions of and Preferences for On-Finger Kinesthetic Notifications. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 518, 17 pages. https://doi.org/10.1145/3544548.3581022

[12] Zhitong Cui, Shuhong Wang, Violet Yinuo Han, Tucker Rae-Grant, Willa Yunqi Yang, Alan Zhu, Scott E Hudson, and Alexandra Ion. 2024. Robotic Metamaterials: A Modular System for Hands-On Configuration of Ad-Hoc Dynamic Applications. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 326, 15 pages. https://doi.org/10.1145/3613904.3642891

[13] Zhitong Cui, Shuhong Wang, Junxian Li, Shijian Luo, and Alexandra Ion. 2023. MiuraKit: A Modular Hands-On Construction Kit For Pneumatic Shape-Changing And Robotic Interfaces. In *Proceedings of the 2023 ACM Designing Interactive Systems Conference* (Pittsburgh, PA, USA) *(DIS '23)*. Association for Computing Machinery, New York, NY, USA, 2066–2078. https://doi.org/10.1145/3563657.3596108

[14] Stefano Dalla Gasperina, Loris Roveda, Alessandra Pedrocchi, Francesco Braghin, and Marta Gandolla. 2021. Review on Patient-Cooperative Control Strategies for Upper-Limb Rehabilitation Exoskeletons. *Frontiers in Robotics and AI* 8 (2021). https://doi.org/10.3389/frobt.2021.745018

[15] Jesús de Miguel-Fernández, Joan Lobo-Prat, Erik Prinsen, Josep M Font-Llagunes, and Laura Marchal-Crespo. 2023. Control strategies used in lower limb exoskeletons for gait rehabilitation after brain injury: a systematic review and analysis of clinical effectiveness. *Journal of neuroengineering and rehabilitation* 20, 1 (2023), 23.

[16] Agostino De Santis, Bruno Siciliano, Alessandro De Luca, and Antonio Bicchi. 2008. An atlas of physical human–robot interaction. *Mechanism and Machine Theory* 43, 3 (2008), 253–270. https://doi.org/10.1016/j.mechmachtheory.2007.03.003

[17] Artem Dementyev, Hsin-Liu (Cindy) Kao, Inrak Choi, Deborah Ajilo, Maggie Xu, Joseph A. Paradiso, Chris Schmandt, and Sean Follmer. 2016. Rovables: Miniature On-Body Robots as Mobile Wearables. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16)*. Association for Computing Machinery, New York, NY, USA, 111–120. https://doi.org/10.1145/2984511.2984531

[18] James Diprose, Bruce MacDonald, John Hosking, and Beryl Plimmer. 2017. Designing an API at an Appropriate Abstraction Level for Programming Social Robot Applications. *Journal of Visual Languages & Computing* 39 (2017), 22–40. https://doi.org/10.1016/j.jvlc.2016.07.005 Special Issue on Programming and Modelling Tools.

[19] Amir Ebrahimi. 2017. Stuttgart Exo-Jacket: An Exoskeleton for Industrial Upper Body Applications. In *2017 10th International Conference on Human System Interactions (HSI '17)*. 258–263. https://doi.org/10.1109/HSI.2017.8005042

[20] Francine Gemperle, Chris Kasabach, John Stivoric, Malcolm Bauer, and Richard Martin. 1998. Design for Wearability. In *Digest of Papers. Second International Symposium on Wearable Computers (Cat. No.98EX215)*. IEEE, 116–122. https://doi.org/10.1109/ISWC.1998.729537

[21] Claire Gordon, Thomas Churchill, Charles Clauser, Bruce Bradtmiller, John McConville, Ilse Tebbetts, and Robert Walker. 1989. Anthropometric Survey of U.S. Army Personnel: Summary Statistics, Interim Report for 1988. (01 1989).

[22] Xiaochi Gu, Yifei Zhang, Weize Sun, Yuanzhe Bian, Dao Zhou, and Per Ola Kristensson. 2016. Dexmo: An Inexpensive and Lightweight Mechanical Exoskeleton for Motion Capture and Force Feedback in VR. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. Association for Computing Machinery, 1991–1995. https://doi.org/10.1145/2858036.2858487

[23] Kelleher R. Guerin, Colin Lea, Chris Paxton, and Gregory D. Hager. 2015. A Framework for End-user Instruction of a Robot Assistant for Manufacturing. In *2015 IEEE International Conference on Robotics and Automation (ICRA '15)*. 6167–6174. https://doi.org/10.1109/ICRA.2015.7140065

[24] Marco Guidali, Alexander Duschau-Wicke, Simon Broggi, Verena Klamroth-Marganska, Tobias Nef, and Robert Riener. 2011. A Robotic System to Train Activities of Daily Living in a Virtual Environment. *Medical & biological engineering & computing* 49 (2011), 1213–1223.

[25] Muhammad Ahsan Gull, Shaoping Bai, and Thomas Bak. 2020. A Review on Design of Upper Limb Exoskeletons. *Robotics* 9, 1 (2020). https://doi.org/10.3390/robotics9010016

[26] Megan Hofmann, Jeffrey Harris, Scott E. Hudson, and Jennifer Mankoff. 2016. Helping Hands: Requirements for a Prototyping Methodology for Upper-limb Prosthetics Users. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 1769–1780. https://doi.org/10.1145/2858036.2858340

[27] Barnabás Homola, Isabella Sheldon, Stela Ago, Milton Mariani, and John Paulin Hansen. 2022. Prototyping Exoskeleton Interaction for Game-Based Rehabilitation. In *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI EA '22)*. Association for Computing Machinery, New York, NY, USA, Article 29, 6 pages. https://doi.org/10.1145/3491101.3503566

[28] Masahiko Inami, Daisuke Uriu, Zendai Kashino, Shigeo Yoshida, Hiroto Saito, Azumi Maekawa, and Michiteru Kitazaki. 2022. Cyborgs, Human Augmentation, Cybernetics, and JIZAI Body. In *Proceedings of the Augmented Humans International Conference 2022* (Kashiwa, Chiba, Japan) *(AHs '22)*. Association for Computing Machinery, New York, NY, USA, 230–242. https://doi.org/10.1145/3519391.3519401

[29] Merel M. Jung and Geke D. S. Ludden. 2018. Potential of Exoskeleton Technology to Assist Older Adults with Daily Living. In *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI EA '18)*. Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/3170427.3188464

[30] Hsin-Liu (Cindy) Kao, Deborah Ajilo, Oksana Anilionyte, Artem Dementyev, Inrak Choi, Sean Follmer, and Chris Schmandt. 2017. Exploring Interactions and Perceptions of Kinetic Wearables. In *Proceedings of the 2017 Conference on Designing Interactive Systems* (Edinburgh, United Kingdom) *(DIS '17)*. Association for Computing Machinery, New York, NY, USA, 391–396. https://doi.org/10.1145/3064663.3064686

[31] Akim Kapsalyamov, Shahid Hussain, and Prashant K Jamwal. 2020. State-of-the-art Assistive Powered Upper Limb Exoskeletons for Elderly. *IEEE Access* 8 (2020), 178991–179001.

[32] Jun Kato, Daisuke Sakamoto, and Takeo Igarashi. 2012. Phybots: A Toolkit for Making Robotic Things. In *Proceedings of the Designing Interactive Systems Conference* (Newcastle Upon Tyne, United Kingdom) *(DIS '12)*. Association for Computing Machinery, New York, NY, USA, 248–257. https://doi.org/10.1145/2317956.2317996

[33] Kazuo Kiguchi, Mohammad Habibur Rahman, Makoto Sasaki, and Kenbu Teramoto. 2008. Development of a 3DOF mobile exoskeleton robot for human upper-limb motion assist. *Robotics and Autonomous Systems* 56, 8 (2008), 678–691. https://doi.org/10.1016/j.robot.2007.11.007

[34] Hubert Kim and Alan T Asbeck. 2021. Just Noticeable Differences for Elbow Joint Torque Feedback. *Scientific Reports* 11, 1 (2021), 23553.

[35] David Ledo, Steven Houben, Jo Vermeulen, Nicolai Marquardt, Lora Oehlberg, and Saul Greenberg. 2018. Evaluation Strategies for HCI Toolkit Research. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–17. https://doi.org/10.1145/3173574.3173610

[36] Yuxuan Lei, Qi Lu, and Yingqing Xu. 2022. O&O: A DIY Toolkit for Designing and Rapid Prototyping Olfactory Interfaces. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 637, 21 pages. https://doi.org/10.1145/3491102.3502033

[37] Nicola Leonardi, Marco Manca, Fabio Paternò, and Carmen Santoro. 2019. Trigger-Action Programming for Personalising Humanoid Robot Behaviour. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300675

[38] Jiahao Li, Jeeeun Kim, and Xiang 'Anthony' Chen. 2019. Robiot: A Design Tool for Actuating Everyday Objects with Automatically Generated 3D Printable Mechanisms. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 673–685. https://doi.org/10.1145/3332165.3347894

[39] Zhuying Li, Tianze Huang, Rakesh Patibanda, and Florian Mueller. 2023. AI in the Shell: Towards an Understanding of Integrated Embodiment. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI EA '23)*. Association for Computing Machinery, New York, NY, USA, Article 28, 7 pages. https://doi.org/10.1145/3544549.3585867

[40] Julio S Lora-Millan, Gabriel Delgado-Oleas, Julián Benito-León, and Eduardo Rocon. 2021. A Review on Wearable Technologies for Tremor Suppression. *Frontiers in Neurology* 12 (2021), 700600.

[41] Azumi Maekawa, Shota Takahashi, MHD Yamen Saraiji, Sohei Wakisaka, Hiroyasu Iwata, and Masahiko Inami. 2019. Naviarm: Augmenting the Learning of Motor Skills Using a Backpack-Type Robotic Arm System. In *Proceedings of the 10th Augmented Human International Conference 2019 (AH '19)*. Association for Computing Machinery, New York, NY, USA, Article 38, 8 pages. https://doi.org/10.1145/3311823.3311849

[42] Marie Muehlhaus, Marion Koelle, Artin Saberpour, and Jürgen Steimle. 2023. I Need a Third Arm! Eliciting Body-Based Interactions with a Wearable Robotic Arm. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 255, 15 pages. https://doi.org/10.1145/3544548.3581184

[43] Florian 'Floyd' Mueller, Nathan Semertzidis, Josh Andres, Joe Marshall, Steve Benford, Xiang Li, Louise Matjeka, and Yash Mehta. 2023. Toward Understanding the Design of Intertwined Human–Computer Integrations. *ACM Transactions on Computer-Human Interactions* 30, 5, Article 73 (sep 2023), 45 pages. https://doi.org/10.1145/3590766

[44] Jun Nishida, Soichiro Matsuda, Hiroshi Matsui, Shan-Yuan Teng, Ziwei Liu, Kenji Suzuki, and Pedro Lopes. 2020. HandMorph: A Passive Exoskeleton That Miniaturizes Grasp. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 565–578. https://doi.org/10.1145/3379337.3415875

[45] Jun Nishida, Yudai Tanaka, Romain Nith, and Pedro Lopes. 2022. DigituSync: A Dual-User Passive Exoskeleton Glove That Adaptively Shares Hand Gestures. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology* (Bend, OR, USA) *(UIST '22)*. Association for Computing Machinery, New York, NY, USA, Article 59, 12 pages. https://doi.org/10.1145/3526113.3545630

[46] C.C. Norkin and D.J. White. 2016. *Measurement Of Joint Motion: A Guide To Goniometry*. F.A. Davis. https://books.google.de/books?id=TSluDQAAQBAJ

[47] Rakesh Patibanda, Elise Van Den Hoven, and Florian 'Floyd Mueller. 2022. Towards Understanding the Design of Body-Actuated Play *(CHI PLAY '22)*. Association for Computing Machinery, New York, NY, USA, 388–391. https://doi.org/10.1145/3505270.3558367

[48] Benoît Pernet, Jacquelyn Kay Nagel, and Hao Zhang. 2022. Compressive Strength Assessment of 3D Printing Infill Patterns. *Procedia CIRP* 105 (2022), 682–687. https://doi.org/10.1016/j.procir.2022.02.114 The 29th CIRP Conference on Life Cycle Engineering, April 4 – 6, 2022, Leuven, Belgium..

[49] Max Pfeiffer, Tim Duente, and Michael Rohs. 2016. Let Your Body Move: A Prototyping Toolkit for Wearable Force Feedback With Electrical Muscle Stimulation. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Florence, Italy) *(MobileHCI '16)*. Association for Computing Machinery, New York, NY, USA, 418–427. https://doi.org/10.1145/2935334.2935348

[50] Ivan Phelan, Madelynne Arden, Maria Matsangidou, Alicia Carrion-Plaza, and Shirley Lindley. 2021. Designing a Virtual Reality Myoelectric Prosthesis Training System for Amputees. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI EA '21)*. Association for Computing Machinery, New York, NY, USA, Article 49, 7 pages. https://doi.org/10.1145/3411763.3443454

[51] L Pignolo, G Dolce, G Basta, LF Lucca, S Serra, and WG Sannita. 2012. Upper Limb Rehabilitation After Stroke: ARAMIS "Robo-mechatronic"' Innovative Approach and Prototype. In *2012 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*. IEEE, 1410–1414.

[52] Jennifer Preece, Helen Sharp, and Yvonne Rogers. 2015. *Interaction Design: Beyond Human-Computer Interaction*. Wiley. https://books.google.de/books?id=n0h9CAAAQBAJ

[53] Tommaso Proietti, Vincent Crocher, Agnès Roby-Brami, and Nathanaël Jarrassé. 2016. Upper-Limb Robotic Exoskeletons for Neurorehabilitation: A Review on Control Strategies. *IEEE Reviews in Biomedical Engineering* 9 (2016), 4–14. https://doi.org/10.1109/RBME.2016.2552201

[54] Jasmine Proud, Daniel Lai, Kurt Mudie, Greg Carstairs, Daniel Billing, Alessandro Garofolini, and Rezaul Begg. 2022. Exoskeleton Application to Military Manual Handling Tasks. *Human Factors* 64, 3 (2022), 527–554. https://doi.org/10.1177/0018720820957467 arXiv:https://doi.org/10.1177/0018720820957467

[55] M. H. Rahman, M. J. Rahman, O. L. Cristobal, M. Saad, J. P. Kenné, and P. S. Archambault. 2015. Development of a whole arm wearable robotic exoskeleton for rehabilitation and to assist upper limb movements. *Robotica* 33, 1 (2015), 19–39. https://doi.org/10.1017/S0263574714000034

[56] Artin Saberpour, Ata Otaran, Martin Schmitz, Marie Muehlhaus, Rishabh Dabral, Diogo Luvizon, Azumi Maekawa, Masahiko Inami, Christian Theobalt, and Jürgen Steimle. 2023. WRLKit: Computational Design of Personalized Wearable Robotic Limbs. In *The 36th Annual ACM Symposium on User Interface Software and Technology* (San Francisco, CA, USA) *(UIST '23)*. Association for Computing Machinery, New York, NY, USA, 13 pages. https://doi.org/10.1145/3586183.3606748

[57] Nihar Sabnis, Dennis Wittchen, Courtney N. Reed, Narjes Pourjafarian, Jürgen Steimle, and Paul Strohmeier. 2023. Haptic Servos: Self-Contained Vibrotactile Rendering System for Creating or Augmenting Material Experiences. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 522, 17 pages. https://doi.org/10.1145/3544548.3580716

[58] Mine Sarac, Massimiliano Solazzi, and Antonio Frisoli. 2019. Design Requirements of Generic Hand Exoskeletons and Survey of Hand Exoskeletons for Rehabilitation, Assistive, or Haptic Use. *IEEE Transactions on Haptics* 12, 4 (2019), 400–413. https://doi.org/10.1109/TOH.2019.2924881

[59] MHD Yamen Saraiji, Tomoya Sasaki, Kai Kunze, Kouta Minamizawa, and Masahiko Inami. 2018. MetaArms: Body Remapping Using Feet-Controlled Artificial Arms. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology (UIST '18)*. Association for Computing Machinery, New York, NY, USA, 65–74. https://doi.org/10.1145/3242587.3242665

[60] Anup Sathya, Jiasheng Li, Tauhidur Rahman, Ge Gao, and Huaishu Peng. 2022. Calico: Relocatable On-cloth Wearables with Fast, Reliable, and Precise Locomotion. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 3 (Sept. 2022), 1–32. https://doi.org/10.1145/3550323

[61] Rafael Sanchez Souza, Filippo Sanfilippo, José Reinaldo Silva, and Arturo Forner Cordero. 2016. Modular Exoskeleton Design: Requirement Engineering with KAOS. In *2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob)*. 978–983. https://doi.org/10.1109/BIOROB.2016.7523756

[62] Michael Suguitan and Guy Hoffman. 2019. Blossom: A Handcrafted Open-Source Robot. *Journal of Human-Robot Interaction* 8, 1, Article 2 (mar 2019), 27 pages. https://doi.org/10.1145/3310356

[63] Richard Tang, Xing-Dong Yang, Scott Bateman, Joaquim Jorge, and Anthony Tang. 2015. Physio@Home: Exploring Visual Guidance and Feedback Techniques for Physiotherapy Exercises. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 4123–4132. https://doi.org/10.1145/2702123.2702401

[64] Shan-Yuan Teng, K. D. Wu, Jacqueline Chen, and Pedro Lopes. 2022. Prolonging VR Haptic Experiences by Harvesting Kinetic Energy from the User. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology (UIST '22)*. Association for Computing Machinery, Article 39, 18 pages. https://doi.org/10.1145/3526113.3545635

[65] Luca Tiseni, Michele Xiloyannis, Domenico Chiaradia, Nicola Lotti, Massimiliano Solazzi, Herman van der Kooij, Antonio Frisoli, and Lorenzo Masia. 2019. On the Edge Between Soft and Rigid: An Assistive Shoulder Exoskeleton with Hyper-redundant Kinematics. In *2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR '19)*. 618–624. https://doi.org/10.1109/ICORR.2019.8779546

[66] Blase Ur, Elyse McManus, Melwyn Pak Yong Ho, and Michael L. Littman. 2014. Practical Trigger-action Programming in the Smart Home. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. Association for Computing Machinery, New York, NY, USA, 803–812. https://doi.org/10.1145/2556288.2557420

[67] David Weintrop, Afsoon Afzal, Jean Salac, Patrick Francis, Boyang Li, David C. Shepherd, and Diana Franklin. 2018. Evaluating CoBlox: A Comparative Study of Robotics Programming Environments for Adult Novices. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3173574.3173940

[68] Nahoko Yamamura, Daisuke Uriu, Mitsuru Muramatsu, Yusuke Kamiyama, Zendai Kashino, Shin Sakamoto, Naoki Tanaka, Toma Tanigawa, Akiyoshi Onishi, Shigeo Yoshida, Shunji Yamanaka, and Masahiko Inami. 2023. Social Digital Cyborgs: The Collaborative Design Process of JIZAI ARMS. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 369, 19 pages. https://doi.org/10.1145/3544548.3581169

[69] Angeliki Zacharaki, Ioannis Kostavelis, Antonios Gasteratos, and Ioannis Dokas. 2020. Safety Bounds in Human Robot Interaction: A Survey. *Safety Science* 127 (2020), 104667. https://doi.org/10.1016/j.ssci.2020.104667

[70] Yang Zhang, Mathieu Bressel, Sander De Groof, Francois Domine, Luc Labey, and Laurent Peyrodie. 2023. Design and Control of a Size-adjustable Pediatric Lower-limb Exoskeleton Based on Weight Shift. *Ieee Access* 11 (2023), 6372–6384.

A Software Implementation

The Arduino firmware was developed with C++ in platformIO. The firmware communicates with the hardware via USB using a serial breakout board and manages the low-level motion control, while the Dynamixel motors internally handle the hardware-level motor control. In a first step, users configure the exoskeleton in the firmware with the `ExoskeletonBuilder`, which follows a builder pattern to add configured joints and define related hardware components and joint ranges, followed by a calibration step which determines the absolute zero-position for each actuated or sensing only joint. These configurations can be persisted in the Arduino's EEPROM for retrieval between microcontroller restarts.

In our event-based architecture, interactive behavior is modeled with `Condition` and `Action` classes. Conditions serve as runtime triggers that control the flow of actions, evaluating to true or false based on the exoskeleton's current state. This evaluation is checked in a control loop that runs at a fixed frequency. Actions encapsulate basic functions and augmentation strategies (cf., Section 4). They can be combined sequentially or in parallel either for individual joints, a set of joints, or the arm for complex control flows. These are managed using `ActionBuilder`, which abstracts the sequentializion and parallelization of actions through another builder pattern, and provides an intuitive programming interface to the user. Additionally, custom commands can be registered to trigger augmentation strategies from the outside through a serial console. The provided command-line interface, GUI, and Processing library build on this capability.

The algorithms for implementing the functional abstraction are provided in the supplemental material. For the full implementation, please refer to the project's GitHub repository[6]

B Hardware Implementation

We designed the hardware components with Autodesk Fusion 360 and fabricated the 3D-printed parts with an Ultimaker S5 printer, using ABS filament. We recommend 100% infill with an octet infill pattern for enhanced stability [48] as well as a normal support structure and 2 mm wall thickness. The modular 3D-printed components are assembled with screws, nuts, and bearings, enabling easy adjustment. The exoskeleton is held in place through a 3D-printed back rod, which is sewn onto a posture corrector. Two additional aluminium rods at the back further stabilize the system.

ExoKit contains only commercially available electronics. It supports the four motors of the Dynamixel XM series that we chose because of their balance between power and cost-efficiency. For sensing, we use Adafruit-based sensors, including load cells with HX711 ADC chips and a panel mount 10K potentiometer for angle tracking. The motors interface with the Dynamixel Shield for Arduino, while an Arduino Mega manages the communication with both the motors and sensors. A Sparkfun serial basic breakout (CH340C) facilitates serial communication with the shield. The motors can be powered with 12 Volt using the Dynamixel power supplies or a portable power bank. For the portable power bank, we leveraged a Zeee Lipo Battery (3300mAh, 14.8V 50C) in a 4S1P configuration (348 grams, 131×43.5×29 mm. We connected the power supply to three voltage converters (12V, 5A, 60 grams) to power the Dynamixel shield and two SMPS2Dynamixel modules inserted into the daisy-chained motors to supply all 6 motors with power. Another voltage converter (5V, 3A, 50 grams) is used to power the Arduino. We put the converters into a box attached to the back of the posture corrector. This resulting setup is completely mobile (see minute 1:30 of the supplemental video).

[6]https://github.com/HCI-Lab-Saarland/ExoKit